INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

Bell & Howell Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

UMI®

Université d'Ottawa · University of Ottawa

3D Object Recognition Using Pseudo-random Color Encoded Structured Light

by

Ziad Sakr, B.A.Sc. in Computer Engineering

A thesis submitted to the
School of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Applied Science
in
Electrical & Computer Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering

School of Information Technology and Engineering
University of Ottawa

Abstract

3D surface detection and measurement is one of the primary tasks involved in machine vision. Visual sensing of 3D position parameters can be done passively or actively.

A method for 3D surface detection and measurement using a projected grid of light is presented. A novel technique using color projected grids is also explained for the detection and extraction of 3D objects in any colored environment. The "point identification" problem is solved using colored pseudo-random multi-valued sequence encoding. A new technique of finding grid line edges instead of its skeleton is also proved to increase the resolution of the extracted object and obtain additional information on its depth from the camera.

Acknowledgements

I would like to thank Dr. Emil M. Petriu, my thesis supervisor, for his time, knowledge and ideas that he provided to me throughout the development of this thesis.

I would also like to thank Dr. Nicholas D. Georganas for providing me with the excellent working facilities and all the necessary equipment required to carry out the work reported in this thesis.

Last, but not least, I would like to thank my parents for their endless love, patience and support, which helped me to complete this work.

List of Acronyms

1D	One Dimensional
2D	Two Dimensional
3D	Three Dimensional
Codec	Encoder/Decoder
GF	Galois Field
GUI	Graphical User Interface
HSL	Hue/Saturation/Lightness
MFC	Microsoft foundation classes
MPEG	Moving Picture Experts Group
PN	Pseudo noise
PRBCS	Pseudo-random binary colored sequence
PRBS	Pseudo-random binary sequence
PRG	Pseudo-random grid
PRMVS	Pseudo-random multi-valued sequence
PRMVCS	Pseudo-random multi-valued colored sequence
PRS	Pseudo-random sequence
RGB	Red/Green/Blue

Table of contents

List of figures

List of tables

Chapter 1

Introduction

The challenge of *computer vision* is trying to extract information of scenes from images using very basic assumptions about physics and optics in order to produce meaningful descriptions of physical objects from images. Image analysis involves the examination of the image data to facilitate solving the vision problem. Image analysis process focuses on two topics: feature extraction and pattern classification. *Feature extraction* is the process of acquiring higher-level image information, such as shape or color information. *Pattern classification* is the act of using this higher-level information to identify objects within the image.

We perceive a world of coherent three-dimensional (3D) objects with many invariant properties. Objectively, the incoming visual data do not exhibit corresponding coherence or invariance; they contain much irrelevant or even misleading variation. Somehow our visual system, from the retinal to cognitive levels, understands, or imposes order on, chaotic visual input. It does so by using information that may reliably be extracted from the input, and also through assumption and knowledge that are applied at various levels in visual processing [27].

There has been a great deal of work on visual sensing. One of the techniques that has been employed for 3D object detection is structured light [1], [3], [6], [8], [12], [34], [36], [37], [40].

Structured light is an efficient technique for obtaining 3D-scene information from single 2D images. It uses a controlled light source for projecting regular patterns (lines or dots),

with a known a priori spatial distribution, to illuminate the scene and hence creates artificial features on the surfaces of objects that are easy to extract. A camera is used to visualize from an angle the structured light projections on object surfaces. The catch here is that the projected light pattern will be uniquely deformed based on the object's shape and orientation. Based on this projected pattern deformation, the 3D object can be detected and its absolute depth calculated.

Problems encountered:

The simultaneous generation of all grid nodes and/or lines may introduce ambiguity in the identification of the individual nodes and/or lines projected on object surfaces, resulting in a so called "point identification" problem [4], which is similar to the "correspondence problem" encountered in stereo vision (see Figure 1.1). Moreover, the color of the environment/scene where the 3D object is placed may cause some problems in extracting the projected grid. The grid might be identified as part of the environment and/or a lot of noise might be present in that extracted grid. In this case, detailed information about the object will be lost.

Solution:

There has been a lot of research work involved to resolve the " point identification" problem encountered. This problem can be solved by identifying the grid lines through color-coding, space encoding, or thickness labeling. But all those methods are limited in their use to a predefined colored environment where the optimum result of the projected grid can be attained.

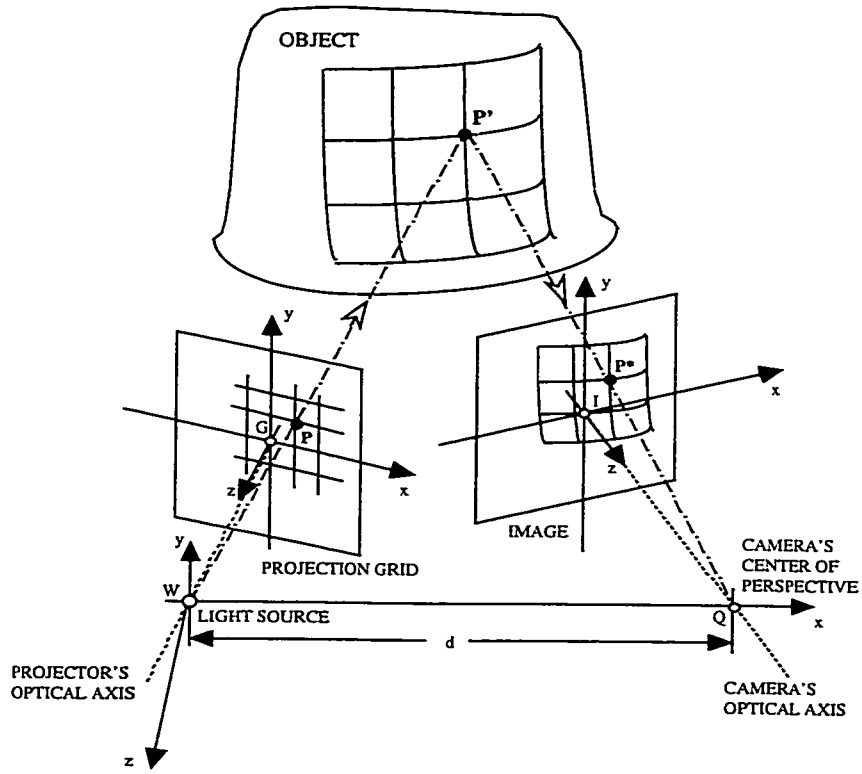


Figure 1.1 "Point identification" in structured light, adopted from [36]

This thesis studies pseudo-random color encoded structured light [36], which is designed to work in a more realistic environment. Different colored grids are projected onto the scene and then the best four grid colors that can be extracted from the scene with least amount of information lost are then selected. A pseudo-random multi-valued sequence "PRMVS" encoded grid (using those four selected colors) is then used to solve the point identification problem. Moreover, a new technique of recovering the edges of the grid lines instead of their skeletons is also implemented, which will yield the four corners instead of the center of each grid node. This approach results in an increase in the density of the points used for triangulation.

This thesis is structured as follows:

Chapter 2 reviews some of the most interesting/important methods used for 3D-object detection from 2D images.

In Chapter 3, a detailed discussion on how to generate pseudo-random sequences "PRS" and their use.

A step-by-step explanation of all the required steps used in this thesis for the detection/extraction of 3D objects from images, including a detailed discussion of the novel techniques proposed and implemented in thesis and their advantages over older techniques is covered in Chapter 4.

In Chapter 5, some experimental results are demonstrated to reveal the success of this thesis.

Finally, the conclusion, contributions and possible further research developments are presented in Chapter 6.

Chapter 2

Techniques for Visual Recognition of 3D Objects

Two-dimensional (2D) images of light intensity do not give explicit information about depth, and hence do not relate directly to a 3D environment. Although there are applications of computer vision which are specifically 2D, there are many other applications which require knowledge of three-dimensional world. Humans are able to infer a great deal of depth information directly from 2D images by stereo vision. The first objective of a 3D vision system is to acquire a depth map, that is a two dimensional array of "pixels" which encodes the depth of the viewed scene element from the point of view of the sensor. In recent years, digitized depth data has been derived from both active and passive techniques. R. A. Jarvis [7] discusses several of those techniques. The active techniques have been dominated by the laser time-of-flight and triangulation. Whereas the principal passive techniques have been: occlusion cues, texture gradient, stereo vision and structured light.

This chapter will focus on 3D-object reconstruction using three passive techniques: occlusion cues, stereo vision, structured light and enhancements made in recent years.

2.1 Occlusion Cues

Rosenberg et al. [22] developed a method for computing the relative depth relationships from occlusion in monocular color imagery. A relaxation labeling [5] process is used to produce a depth map, which is used to test the consistency of a depth graph derived from occlusion cues. A segmented image is used as input – each region is assumed to be distinguished from adjacent regions on the basis of the primary features of color and

texture. It is assumed that if some of these regions represent only parts of objects, this is purely the result of occlusion effects. Occlusion evidence is obtained by examining clusters of adjacent regions, each cluster being evaluated in terms of different occlusion cases. Where there are distinct occlusion related groupings with no occlusion clues between the groupings, relative depth relationship can be determined using probabilistic relaxation labeling [5]. This labeling technique resolves and reduces the possible contradictions in local occlusion data and establishes relationship between different regions based on their depth.

The main weakness in this technique though, is the proper segmentation of the object image. The lack of proper segmentation obtained from depth maps and occlusion cues would produce false results.

2.2 Stereo Vision

In the human visual system, it is believed that depth recovery does not need the existence of high level information. Humans recover relative depth quite easily, but fail to recover absolute depth. Computers on the other hand, have difficulty in recovering both relative and absolute values.

If one tries holding his/her finger up in front of him/her and slowly blinking his/her eyes alternately, he/she will notice his/her finger shifting by a certain amount in each view. Moving his/her finger away from his/her face reduces the shift, whereas the shift increases as the finger gets closer. This shift is called *disparity* and is related to the distance between the viewer and the feature being viewed [27]. In fact, if the fixation point of the two viewpoints is at infinity, then disparity is inversely proportional to depth. By comparing the relative disparities of corresponding features in two images a depth recovery process can be initiated to some form of three-dimensional reconstruction of the viewed scene.

2.2.1 Stereo Geometry

For the pinhole camera model, also known as the perspective camera model, light rays from a point in the real world travel in a straight line through the optical center of the

camera and are projected on the view plane (Figure 2.1). This is a projection from 3D space onto 2D space through a single point.

For two cameras displaced by a rotation and translation (Figure 2.2), the images are related by another projective transform, known as the *Fundamental Matrix* [38]. In stereo applications, the fundamental matrix can be derived in two ways, either by knowing the calibration and displacement of the cameras, or by finding eight or more matching points.

All rays imaged by the first camera originate from the optical center of the first camera, and the same can be said for the second camera. The line that connects these two optical centers is known as the *baseline* of the cameras, and the two points at which the baseline intersects the view planes are known as the *epipoles* (Figure 2.3). For some point X in space, the plane defined by the epipoles and the point X is the *epipolar plane* for that point. Its images x and x' will lie in the intersection between the viewplanes and the epipolar plane, along the lines known as the *epipolar lines* [39].

As the point moves toward and away from the cameras, its image moves along the epipolar line. In fact, all points which exits in one epipolar line will also exist in the other epipolar line. For cameras where the Fundamental Matrix has been derived, this reduces the search for corresponding points to a one-dimensional search.

An even simpler configuration occurs when the viewplanes of the two cameras are parallel to each other, and their translation consists of only a horizontal displacement. Then the epipoles exist, effectively, at infinity and the epipolar lines are parallel to each other. This simplifies the search matches, especially for digital images, to matches within a scanline. The equation for the distance of points simplifies to $Z = BfD$ up to a scale factor (see Figure 2.4). Many stereo algorithms assume this simplifying configuration.

2.2.2 *Computational Stereo Paradigm*

The objective of disparity based computer stereo vision is to determine the location of objects (or surfaces) in a scene relative to a camera or world coordinates. This process can be divided into three general steps.

2.2.2.1 Acquisition of images

For disparity methods to work, an image must be displaced somehow relative to the other image. There is a trade-off that should be considered. A small separation between

cameras makes the correspondence problem (will be discusses in the next section) easier, but impairs the resolution and hence the depth accuracy. A large camera separation will allow accurate depth reconstruction but makes feature correspondence difficult, since the same features are widely separated in the two images [13].

2.2.2.2 Feature matching: the correspondence problem

Establishing a correspondence between points in stereo images is the most important and difficult step towards the reconstruction of a 3D scene from a stereo image. The camera geometry (discussed earlier) can affect greatly the amount of processing required for feature matching/extraction. Several matching techniques between images have been used [14]. Dhond and Aggrawal [2] have described the difference between area-based and feature based matching techniques.

2.2.2.2.1 Area-based techniques

Area-based stereo techniques use correlation among brightness (intensity) patterns in the local neighborhood of a pixel in one image with the brightness patterns in a corresponding neighborhood of a pixel in the other image. First, a point of interest is chosen in one image. A cross-correlation measure is then used to search for a point with a matching neighborhood in the other image. The area-based techniques have a disadvantage in that they use intensity values at each pixel directly, and are hence sensitive to distortions as a result of changes in viewing position as well as changes in absolute intensity, contrast and illumination. Also the presence of occluding boundaries in the correlation window tends to confuse the correlation-based matcher, often giving erroneous depth estimate.

2.2.2.2.2 Feature-based techniques

Feature-based stereo techniques use symbolic features derived from intensity images rather than image intensities themselves. Hence, these systems are more stable towards changes in contrast and ambient lighting. The features used most commonly are either edge points or edge segments (derived from connected edge points) that may be located with subpixel precision. Also feature-based methods allow for simple comparisons between attributes of the features being matched.

2.2.2.3 Depth map calculations

Finally, the shift, or disparity, between each pair of matched features is calculated. The distance of the feature to the camera is a function of the disparity between the matching features and the intrinsic and extrinsic parameters of the two cameras. If these parameters are known then the cameras are said to be calibrated. The camera parameters do not necessarily need to be calculated ahead of time, a few robust feature matches are sufficient to determine the fundamental matrix.

Two basic problems might arise using the stereo disparity technique in retrieving depth information about a 3D object. The first arises at parts of the image where uniformity of intensity or color makes matching impossible. The second might occur when the image of some part of the scene appears in only one view of a stereo pair because of occlusion effects or because of the limited field of view captured in the images.

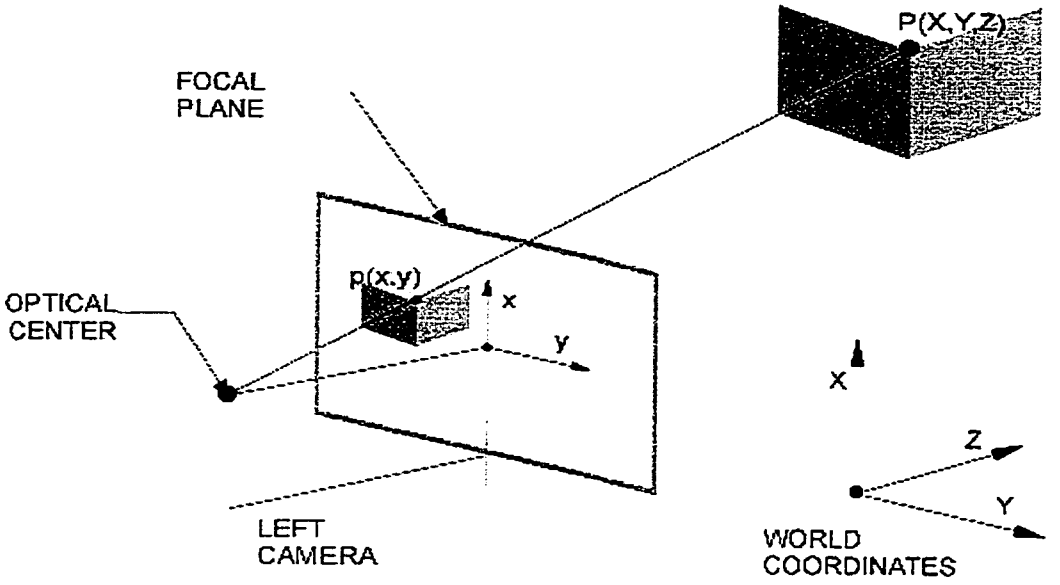


Figure 2.1 Pinhole camera model

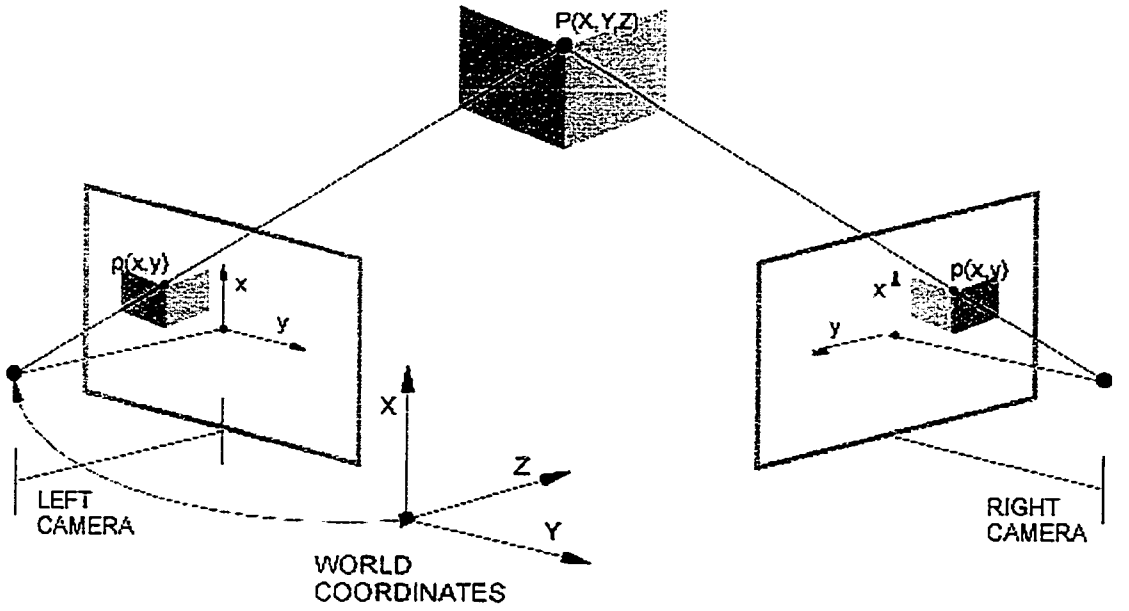


Figure 2.2 Stereo camera configuration

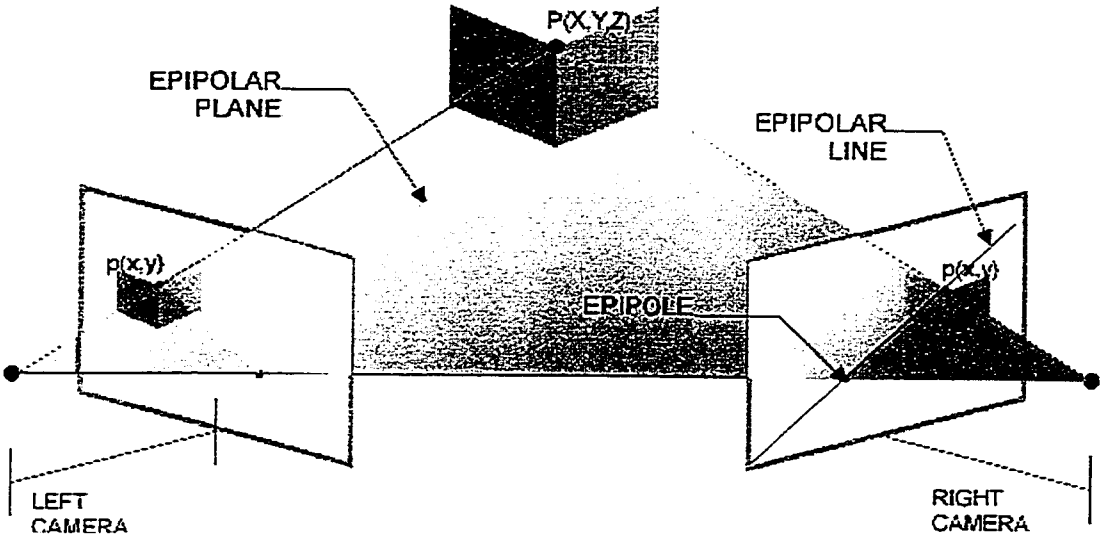


Figure 2.3 Epipolar geometry of stereo cameras

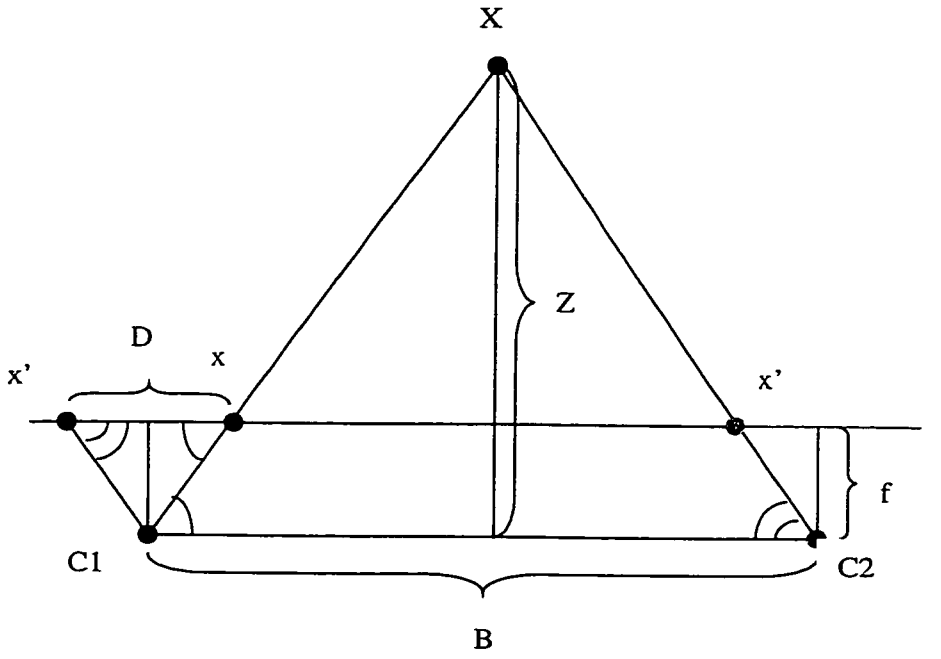


Figure 2.4 Parallel stereo cameras

2.3 *Structured Light*

Structured light is one efficient approach for acquiring 3D information about the world. It is a general concept and there are a number of ways of exploiting it to obtain 3D data. The structured light system consists of a projector and a camera. The projector produces a beam or several beams of light, which are projected onto the world and captured by the camera. Hence for each visible point projected onto the world there is a unique image location (pixel coordinates).

The virtue of structured light is that it replaces natural features (object edges, for example), if any, on object surfaces in the scene by artificial features that are very easily detected. False features caused by noise under ambient illumination are eliminated in the sense that the noise would not be present when structured light is employed where only bright light spots (or lines or grids) are visible to the camera and elsewhere is assumed totally dark. Thus a binary image is easily created. Difficulties due to lack of features in images are overcome since structured light always creates features on surfaces in the scene. The missing feature problem is also partly solved because the matching carried out here is from the camera image to the projector "image", and features (bright light strips) in the projector image are complete. In other words, if a light stripe appears in the camera image, the corresponding grid line must also appear in the source that originates the stripe – in the projector image. However, due to occlusion, not all stripes projected will be visible to the camera. Thus under structured light, the problems inherent in stereo matching are partly removed.

One of the controlled light schemes is to use a single ray of light scanned across the scene, producing a single light stripe for each position (Figure 2.5). When the light source is displaced from a viewing camera, the camera view of the stripe shows displacement along the stripe, which is proportional to depth; a kink indicates a change of plane and a discontinuity, a physical gap between surfaces. The proportionality constant between the beam displacement and depth is dependent upon the displacement of the source from the camera so that more accurate depth measurements can be made with larger displacements.

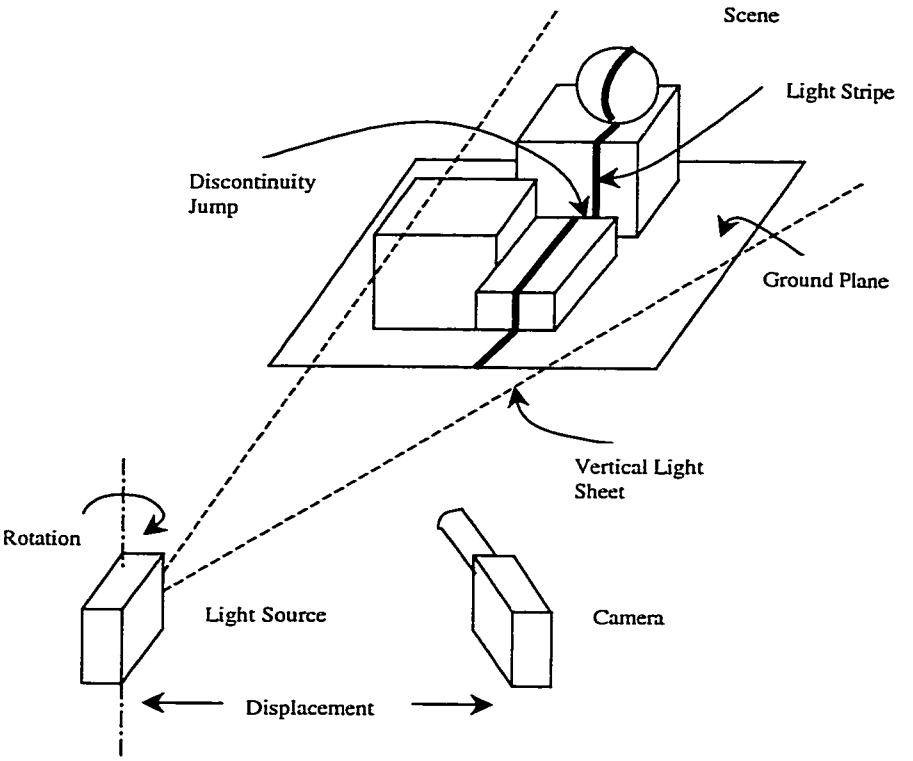


Figure 2.5 Structured light using stripe lighting [7]

Since a single light scheme requires a successive scan over the scene, it will take longer than necessary to process those lines for 3D reconstruction.
An entire grid of lines can be projected on the scene at once. The grid of lines may be two perpendicular sets of equally spaced parallel lines on a projector slide (Figure 2.6).

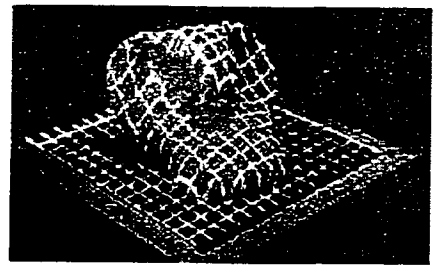


Figure 2.6 Projected grid line onto a scene

Will and Pennington [1] described a method by which the locations and orientations of planar areas of polyhedral solids are extracted through linear frequency domain filtering applied to images of scenes illuminated by a high contrast rectangular grid of lines. Edges are defined through the intersections of the extracted planes. Similar method was proposed by Wang and al [3] for 3D-object extraction. The idea was later enhanced by Hu and Stockman [4], where a connected network of lines is processed after being obtained from projecting a grid of lines onto a scene and removing the background. Through a calibrated camera, the intersecting points are analyzed and depth calculated. The extracted network is also tested for any disconnected lines, which might provide some information on the number of objects present in the scene.

As stated earlier, the correspondence problem in stereo matching is to identify feature-pairs in two images that correspond to the same physical surface feature. Using structured light, the feature points are those pixels made bright by the structured light. If a single light beam creates a single bright dot in the image or a single light plane creates a single

bright stripe curve in the image [6], the correspondence problem is immediately solved – the dot or the stripe curve is the only matching candidate. Since a single light scheme requires a successive scan over the scene, it will be slower than necessary. An entire grid of lines can be projected onto the scene at once. But now the problem is which line on the slide is the line that generated the stripe on the scene. Thus, the correspondence problem becomes the *line-labeling problem.* This problem can be solved by identifying the grid lines through color coding[8], space encoding [15][9], thickness labeling [9], and pseudo random binary sequences[21].

Boyer and Kak [8] developed a method for extracting 3D information of a scene by projecting a color encoded structured light to avoid the line identification problem. But this method has two problems. First, a priori knowledge of the color of each line must be known before hand. This requires a certain amount of computer memory before the processing begins. The second and most important problem is that the lines should be projected onto a scene with color content, which is predominantly neutral and does not affect the recovery of the projected stripe by the camera. This is almost impossible to apply in a natural environment where the color might vary from one part to the other in the scene. The error generated from noise in the images and the recovery and reconstruction of the grid line network will be a major factor to consider.

All these factors are taken into consideration in this thesis, which studies the performance of color encoded structured light combined with pseudo-random multi-valued sequence encoding [40]. An opportunistic real-time grid color assignment is used in order to increase the efficiency of the color encoded structured light in any colored environment/scene. All detailed explanation of this work is presented in Chapter 4.

Chapter 3

Pseudo-Random Coding

Pseudo-random coding has been a very efficient technique for absolute position recovery applications [18], [19], and [20]. This thesis discusses applications of pseudo-random encoding for 3D object identification from a 2D image. But before proceeding into the details implemented in this thesis, a general understanding of pseudo-random sequences, how they are generated and some of their useful properties will be discussed in this chapter.

3.1 Pseudo-Random Sequences

Because the theory of pseudo-random sequences is based on the theory of finite fields, a short review of finite fields is first given.

3.1.1 Finite (Galois) Fields

A field is a set of elements **F** equipped with two binary operations "+" (addition) and "x" (multiplication) such that for all a, b, c $\in$ **F** the following axioms hold [10], [25], [26], [32]:

a) **F** is an abelian group under addition ("+"):

1) Addition in **F** is commutative

$$a+b = b+a;$$

2) Addition in F is associative

$$a+(b+c) = (a+b)+c;$$

3) There exists an element *0* belonging to **F** such that

$$a+0 = 0+a = a;$$

for all a belonging to **F**.

4) For each element a belonging to **F**, there exists an element $-a$ belonging to **F** such that

$$a+(-a) = (-a)+a = 0;$$

b) **F** is an abelian group under multiplication ("x"):

1) Multiplication in **F** is commutative

$$a\text{x}b = b\text{x}a;$$

2) Multiplication in **F** is associative

$$a\text{x}(b\text{x}c) = (a\text{x}b)\text{x}c;$$

3) There exists an element 1 belonging to **F** such that:

$$a\text{x}1 = 1\text{x}a = a;$$

for all a belonging to **F**.

4) For each element a belonging to **F**, there exists an element a^{-1} belonging to **F** such that:

$$a\text{x}(a^{-1}) = (a^{-1})\text{x}a = 1;$$

c) The following distributive laws hold:

1) $(a+b)\text{x}c = (a\text{x}c)+(b\text{x}c)$;

2) $a\text{x}(b+c) = (a\text{x}b)+(a\text{x}c)$;

The symbol 0 is the identity element of the additive group, and the symbol 1 is the identity element of the multiplicative group.

The rational numbers **Q**, real numbers **R**, and complex numbers **C** are examples of fields. Because each one of these fields contains an infinite number of elements, they are called *infinite fields*. A field that contains a finite number of elements is called a *finite field* [49]. The number of elements in a field is called the order of the field. A field with q elements is called a *Galois field* and is denoted by GF(q) [50] [51].

3.1.1.1 Construction of finite fields

The simplest class of finite fields is this with prime number of elements. This class of fields is important because any finite field contains a subfield belonging to this class.

A field belonging to this class can be constructed in the following manner:

> Let p be any prime number. The integers under addition and multiplication modulo p form a field GF(p) with p elements. The elements of these fields are {0, 1, 2, ..., p-1}.

For example, if p=5 the operations for addition and multiplication are given in Table 3.1 and Table 3.2 respectively:

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

Table 3.1 Addition table for GF(5)

x	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

Table 3.2 Multiplication table for GF(5)

An important property of this class of fields is that the elements of the field, except the element *0* (the identity element for addition), form a cyclic group under multiplication. This means that any finite field with a prime number of elements contains at least one element called a generator from which all the other elements of the field can be obtained. For example, the element 2 for GF(5) is a generator element for the cyclic group {1, 2, 3, 4} under multiplication modulo 5:

$$2^1 = 2;\ 2^2 = 4;\ 2^3 = 3;\ 2^4 = 1;$$

The class of finite fields that has a number of elements equal to a power of a prime is the most general class of finite fields. These classes of fields are known as polynomial fields, or Galois fields of order $q=p^m$, and are denoted by GF(p^m).

A polynomial field with $q=p^m$ elements, where p is a prime and m is any positive integer, can be constructed as follows:

- the elements of the field are all polynomials in x of degree less than or equal to m-1 with coefficients from GF(p).
- addition is done in the ordinary way, modulo p.
- multiplication is done by multiplying modulo p, and taking the remainder when divided by a polynomial h(x).

The polynomial h(x) is any irreducible polynomial with coefficients from GF(p) of degree m. Recall that an irreducible polynomial is a polynomial with coefficients from GF(p) which is not the product of two polynomials of lower degree.

The field constructed in this way contains p^m elements and is denoted by GF(p^m). It can be proven that a finite field of order p^m exists for all primes p and positive integers m, and that these are the only finite fields that exist.

Consider the following example:

$p = 3$, $m = 3$, $q = p^m = 27$, $h(x) = x^3+2x+1$

The elements of the field GF(27) are:

0, 1, 2, x, x+1, x+2, 2x, 2x+1, 2x+2, x^2, x^2+1, x^2+2, x^2+x, x^2+x+1, x^2+x+2, x^2+2x, x^2+2x+1, x^2+2x+2, $2x^2$, $2x^2$+1, $2x^2$+2, $2x^2$+x, $2x^2$+x+1, $2x^2$+x+2, $2x^2$+2x, $2x^2$+2x+1, $2x^2$+2x+2.

The coefficients of these polynomials are obtained by taking all possible three-element combinations of the elements of GF(3).

An example of polynomial addition in this field is as follows:

$(x^2+x+1) + (x^2+2x) = 2x^2+1$;

$(x+1) + (2x^2+2x+2) = 2x^2$;

$(2x^2+2) + (2x^2+x) = x^2+x+2$;

An example of polynomial multiplication is as follows:

$$(x^2+2x)(2x^2+1) = 2x^4+x^3+x^2+2x$$
$$= (x^3+2x+1)(2x+1) + (x+2)$$
$$= h(x)(2x+1) + (x+2)$$

$\rightarrow (x^2+2x)(2x^2+1) = (x+2)$ when $h(x) = x^3+2x+1$

The elements of any finite field can be expressed in four different ways:

- as a m-tuple of elements belonging to GF(p)
- as a polynomial of degree lower or equal to m with coefficients from GF(p)
- as a power of t (where t is a primitive element, a zero for h(x))
- as a logarithm

The irreducible polynomial h(x) used to generate a field is called the generator polynomial. It has been proven that for any positive integer m and any prime p, there exists a primitive polynomial of degree m with coefficients from GF(p) [25].

The non-zero elements of the field form a cyclic group of order p^m-1 with generator t. Any generator for this cyclic group is called a primitive element for GF(p^m). Any irreducible polynomial having a primitive element as a zero is called a primitive polynomial. Not all irreducible polynomials are primitive. All finite fields contain a primitive element. Usually, the minimal polynomial of a primitive element of GF(p^m) has degree m and is chosen to be the generator polynomial for GF(p^m). Tables of primitive polynomials for some m and p can be found in [26] [17].

The correspondence between the first two representations (m-tuple or polynomial) and the last two representations (power of t or logarithm) can be found as previously shown by dividing the polynomial x^n by the primitive polynomial h(x) and taking the remainder (note that t is zero for h(x)).

Table 3.3 gives the elements of GF(27) expressed in all four possible ways. The element t^4, for example, can be written as:

$$x^4 = (x^3+2x+1)x+x^2+2x$$

so the correspondence is given by:

$$t^4 \rightarrow 120$$

000	0	0	$-\infty$
001	1	1	0
010	x	t	1
100	x^2	t^2	2
012	x+2	t^3	3
120	x^2+2x	t^4	4
212	$2x^2+x+2$	t^5	5
111	x^2+x+1	t^6	6
122	x^2+2x+2	t^7	7
202	$2x^2+2$	t^8	8
011	x+1	t^9	9
110	x^2+x	t^{10}	10
112	x^2+x+2	t^{11}	11
102	x^2+2	t^{12}	12
002	2	t^{13}	13
020	2x	t^{14}	14
200	$2x^2$	t^{15}	15
021	2x+1	t^{16}	16
210	$2x^2+x$	t^{17}	17
121	x^2+2x+1	t^{18}	18
222	$2x^2+2x+2$	t^{19}	19
211	$2x^2+x+1$	t^{20}	20
101	x^2+1	t^{21}	21
022	2x+2	t^{22}	22
220	$2x^2+2x$	t^{23}	23
221	$2x^2+2x+1$	t^{24}	24
201	$2x^2+1$	t^{25}	25

Table 3.3 Elements generated by $h(x) = x^3+2x+1$, expressed in four different ways

An element t can be obtained from the previous one t^{n-1} by multiplying it with t. If the result is a polynomial with a degree greater than m the same procedure is applied, that is the result is divided by h(x) and only the remainder is taken. For example

$$x^{18} = x^{17}x = (2x^2+x)x = 2x^3+x^2 = h(x)x+(x^2+2x+1)$$

More tables found in Appendix B give the elements of the Galois fields: GF(2), GF(2^2), GF(2^3), GF(2^4), GF(2^5), GF(2^6), GF(3) and GF(3^2).

3.1.2 Feedback shift register and PRS generation

Pseudo-random sequences (PRS), which are also called pseudo-noise (PN), maximal-length shift-register sequences, or m-sequences [43] [44] [45] [46] [47] [48], are sequentially generated as the output of a shift register.

For any given primitive polynomial h(x) of degree m with coefficients from GF(p), a feedback shift register can be constructed as follows:

- the number of memory cells is equal to m
- each memory cell contains an element of the field GF(p)
- at each time unit, the contents of the memory cells are shifted one place to the right
- the content of the left most memory element is the sum of the terms corresponding to h(x) (all operations are done in GF(p)).

Suppose

$$h(x) = x^m + h_{m-1}x^{m-1} + \ldots + h_1x + h_0$$

with $h_i \in$ GF(q), $h_0 \neq 0$. The shift register specified by h(x) is shown in Figure 3.1.

In this figure the boxes contain elements of GF(q), say a_{i+m-1}, ..., a_i. The feedback path then forms

$$a_{i+m} = -h_{m-1}a_{i+m-1} - h_{m-2}a_{i+m-2} - \ldots - h_1 a_{i+1} - h_0 a_i$$

which is the recurrence describing the output sequence. This is an infinite sequence of period q^m-1 (if the starting state is not zero), and each nonzero state appears once in a period. A segment of the output sequence of length q^m-1 is called a pseudo-random sequence over GF(q). Let $\delta_m(q)$ be the set of such sequences, together with 0.

For example, if $q = 2^2$, $m = 2$, $h(x) = x^2 + x + w$ with $GF(2^2) = \{0, 1, w, w^2\}$ and $w^2 + w + 1 = 0$, $w^3 = 1$, the following pseudo-random sequence is obtained of length 15:

$$0 \quad 1 \quad 1 \quad w^2 \quad 1 \quad 0 \quad w \quad w \quad 1 \quad w \quad 0 \quad w^2 \quad w^2 \quad w \quad w^2$$

Then $\delta_2(2^2)$ consists of 0 and the 15 cyclic shifts.

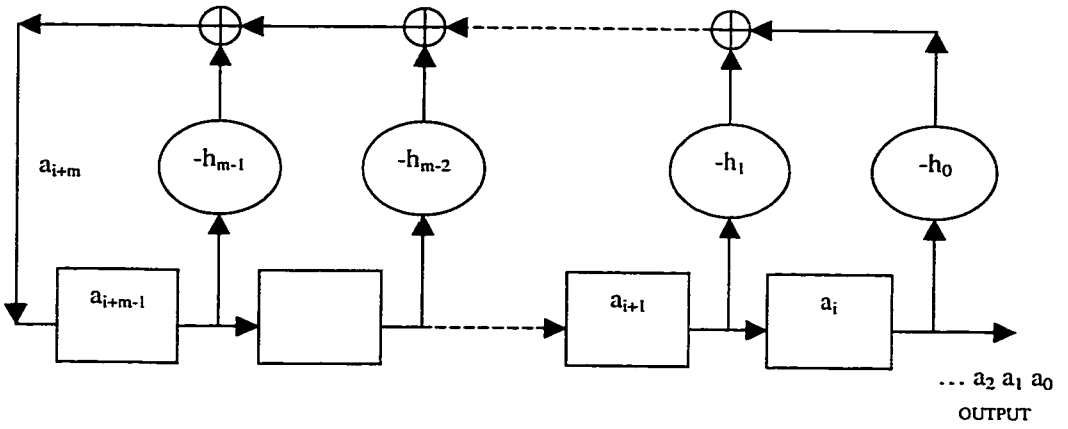


Figure 3.1 Shift register specified by h(x)

The correspondence between an element of the field expressed as m-tuple and the same element expressed as a logarithm can be obtained using a modified feedback shift register as further explained (see Figure 3.4). The states of the feedback shift register represent an index list, a list of the elements of the field in m-tuple form. The number of steps needed to arrive at any given state represents the logarithm expression of the Galois field element corresponding to the given state.

Figure 3.2 shows the feedback shift register that gives GF(3) when m = 3 and h(x) = x^3+2x+1. The corresponding feedback equation is:

$$x_{i+3} = x_{i+1}+2x_i$$

and $x_2 = 1$, $x_1 = 0$ and $x_0 = 0$ is an arbitrary initial state.

To obtain the correspondence between the elements of the Galois field (Table 3.3), the modified feedback shift register, shown in Figure 3.3, has to be used. It represents the reverse of the shift register presented in Figure 3.2. The states of this shift register provide a list of the elements of $GF(3^3)$, both as successive powers of t and as m-tuple or polynomial in x of degree at most three, where only the coefficients of the polynomials are given. The next example presented in Figure 3.4 illustrates the generation of PRS for GF(27) with 001 as the initial m-tuple state. The states of the modified feedback shift register and the regular feedback shift register are also given. The contents of the cells of the modified feedback shift register are obtained according to the following equations, which result directly from Figure 3.3:

$$(\text{cell } 0)_k = (\text{cell } 1)_{k-1}$$
$$(\text{cell } 1)_k = (\text{cell } 0)_{k-1} + (\text{cell } 2)_{k-1}$$
$$(\text{cell } 2)_k = 2 * (\text{cell } 0)_{k-1}$$

As shown in Figure 3.4, all the elements of $GF(3^3)$ appear in this development just once. Each line is identical with the previous one with the exception that the last symbol of the sequence is obtained as the output of the modified feedback shift register, and the state of

the register is obtained according to the equation presented above. An index can be associated with each line, which represents the number of steps required for a shift register to arrive at a specific state starting from the original state 001. These indexes are in fact the logarithm representation of the elements of the Galois field.

For each finite field GF(p^m) there are p^m-1 PRSs that are basically the same sequence shifted to the right or left by a number of symbols depending on the state of the register when the sequence generation was started.

For all our next discussion, we consider for each field only the sequence that starts with the element corresponding to the primitive element of the field. This sequence has the property that ensures the correspondence between the two forms of expressing the elements of the Galois field (the polynomial form and the logarithmic form). By using this sequence, pseudo-random to natural decoding is equivalent to finding the corresponding logarithm representation for a given element known in its polynomial representation.

Since there are m memory elements containing numbers between 0 and $p-1$, there are p^m possible states for the shift register. Thus, the infinite sequence that the shift register will generate is periodic. The state in which all the memory elements are zero is not included because the shift register cannot leave this state. Hence, the maximum length of the sequence is p^m-1. This is the reason why the pseudo-random sequences are also called maximum-length shift register sequences.

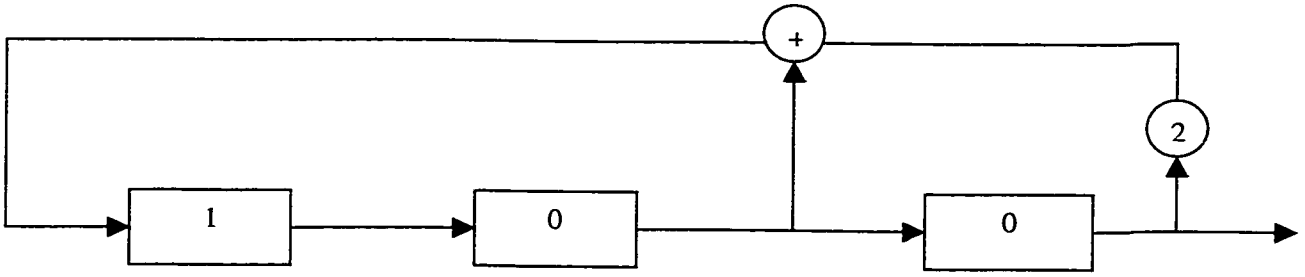


Figure 3.2 Feedback shift register for GF(27) with $h(x) = x^3+2x+1$

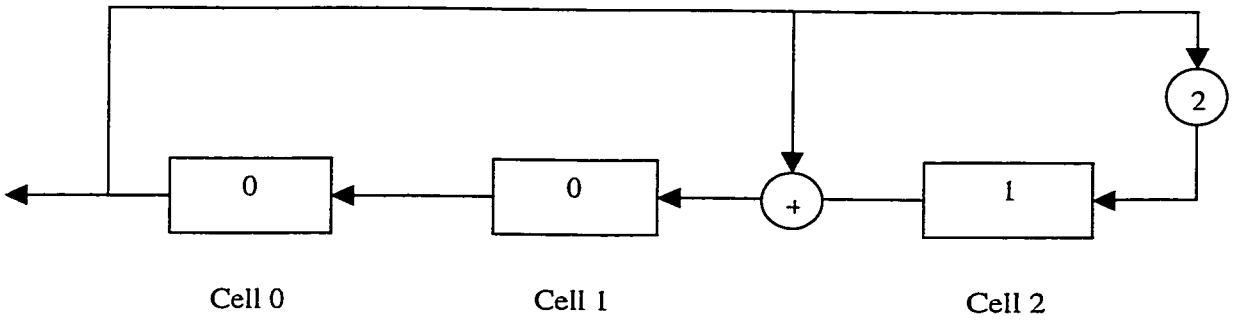


Figure 3.3 Modified feedback shift register for GF(27)

Clock step	PRS	State of the modified feedback shift register	State of the feedback shift register
0	0	001	001
1	00	010	010
2	001	100	101
3	0010	012	012
4	00101	120	121
5	001012	212	211
6	0010121	111	112
7	00101211	122	120
8	001012112	202	201
9	0010121120	011	011
10	00101211201	110	111
11	001012112011	112	110
12	0010121120111	102	100
13	00101211201110	002	002
14	001012112011100	020	020
15	0010121120111002	200	202
16	00101211201110020	021	021
17	001012112011100202	210	212
18	0010121120111002021	121	122
19	00101211201110020212	222	221
20	001012112011100202122	211	210
21	0010121120111002021221	101	102
22	00101211201110020212210	022	022
23	001012112011100202122102	220	222
24	0010121120111002021221022	221	220
25	00101211201110020212210222	201	200

Figure 3.4 Generation of PRS over GF(3^3) of length 3^3-1

3.1.2.1 Properties of PRS

Let h(x) be a fixed primitive polynomial of degree m, and let $\delta_m(q)$ be the set consisting of the pseudo-random sequences obtained from h(x). The following properties of these sequences apply [17]:

Property 1. The Shift Property: If $b=b_0b_1\ldots b_{q-2}$ is any PRS in $\delta_m(q)$, then any cyclic shift in b, say

$$b_jb_{j+1}\ldots b_{q-2}b_0\ldots b_{j-1}$$

is also in $\delta_m(q)$.

Property 2. The Recurrence Property: Any PRS $b\in \delta_m(q)$ satisfies the recurrence

$$b_{i+m} = h_{m-1}b_{i+m-1}+h_{m-2}b_{i+m-2}+\ldots+h_1b_{i+1}+b_i \qquad i = 0, 1, \ldots$$

Property 3. The Window Property: If a window of width m is slid along a PRS in $\delta_m(q)$, each of the p^m-1 non-zero binary m-tuple is seen exactly once. This follows from the fact that h(x) is a primitive polynomial. Let us consider the PRS generated by the primitive polynomial $h(x) = x^3+2x+1$ over GF(3). As previously shown a cycle of this sequence is given by:

00101211**201**1100202122102 22

The three bolded symbols form a window that is unique in this sequence.

Note: to avoid any difficulties at the ends of the sequence, imagine that the PRS is written in a circle. That is, after the last digit in the sequence comes the first digit in the sequence.

Property 4. The Pseudo-Random Property: In any PRS in $\delta_m(q)$, *0* occurs q^{m-1}-1 times and every non-zero element of GF(q) occurs q^{m-1} times.

Property 5. The Addition Property: The sum of two sequences in $\delta_m(q)$ (formed component-wise, modulo m, without carries) is another sequence in $\delta_m(q)$. For instance, the sum of the first two sequences given above in the set of 26 sequences is the tenth sequence.

Property 6. The Shift-and-Add Property: The sum of a pseudo-random sequence and a cyclic shift of itself is another pseudo-random sequence.

Property 7a: A PRS in $\delta_m(q)$ has the form

$$a=b,\ \gamma b,\ \gamma^2 b,\ \ldots,\ \gamma^{q-2} b,$$

where b is a sequence of length $(q^m-1)/(q-1)$ and γ is a primitive element of GF(q). This is because the states of the shift register can be made to correspond to a logarithmic table of GF(q).

For example, if $q = 2^2$, $m = 2$ and $h(x) = x^2+x+w$, the following pseudo-random sequence is obtained:

$$011w^2 10ww1w0w^2w^2ww^2$$

In this case $b= 011w^2 1$ and $\gamma = w$.

Property 7b: Let $a = (a_0 \ldots a_{n-1})$ be a PRS in $\delta_m(q)$, and let $b =(a_s a_{s+1} \ldots a_{s-1}) = (b_0 \ldots b_{n-1})$ be a shift of a by s places. i) If s is not a multiple of q-1, then among the q^m-1 pairs (a_i,b_i), (0, 0) occurs q^{m-2}-1 times and every other pair of elements of GF(q) occurs q^{m-2} times; ii) If $s = j(q-1)$, then (0, 0) occurs q^{m-1}-1 times and the pairs $(\alpha, \gamma^j \alpha)$, for all nonzero α in GF(q), occur q^{m-1} times. In particular, if s = 0 (no shift) each (α, α), $\alpha \neq 0$, occurs q^{m-1} times. For example, if the PRS given in property 7a is shifted once,

thus

$$011w^2 10ww1w0w^2w^2ww^2$$

$$11w^2 10ww1w0w^2w^2ww^2 0$$

then (0, 0) does not occur and every other pair (0, 1), (0, w), ..., (w^2, w^2) occurs once.

3.1.3 1D Pseudo-Random Window Index Recovery

Let h(x) be a polynomial of degree m with coefficients from GF(p), and a$\in \delta_m(q=p^m)$, a PRS is generated using a feedback shift register as explained earlier. Let us also consider a m-size window belonging to this sequence, represented using bold characters below, where the element a_k is considered to be the origin of the window.

$$a= a_0a_1a_2 \ldots \mathbf{a_k a_{k+1} a_{k+2}} \ldots \mathbf{a_{k+m-1}} a_{k+m} \ldots a_{q-2}$$

The problem of window index recovery can be stated as follows:

Given the elements of an m-size window belonging to a PRS generated by h(x) over GF($q=p^m$) determine the k index of the origin of the window a_k.

Due to the fact that a window represents an element of GF(q) in its m-tuple form, the problem can be stated as determining the logarithm representation of a given element of the field when the m-tuple form is known.

This translation is always necessary for practical applications that use PRSs for encoding. As we will see in the next section, 2D pseudo-random window index recovery is also reducible to a 1D pseudo-random window index recovery.

Few methods known in literature that can be applied for solving that problem:

- A serial-type code conversion algorithm, extensively discussed in [31] exploits the reversibility of the PRS generating algorithm. This method is based on the idea that it is possible to find the index (the logarithm

representation) associated with any pseudo-random m-tuple by simply counting the number of reverse feedback shifts that it takes for the given m-tuple to shift back into the initial state of the shift register.

- A strictly parallel solution would be to use a code conversion table stored in ROM. This is expensive for applications requiring high encoding resolutions.

- A compromise [18] is a combination of the serial and parallel methods as exemplified in Figure 3.5. Consider a pseudo-random encoding track where certain positions (uniformly distributed with a period of t) are employed as "milestones". The code conversion for any position p=m*t+r, where m*t is the position of the nearest "down the track milestone", Q(m) and r represent the position relative to this milestone, will be discussed. The natural code for r is found by counting the steps required to arrive by successive back-shifts from the initial code to the nearest milestor Q(m). All intermediate states of this serial shift-back operation are checked in parallel against all possible milestone pseudo-random patterns. Thus with this method, the code conversion of the relative position r distance is found serially while the milestone code conversion is done in parallel.

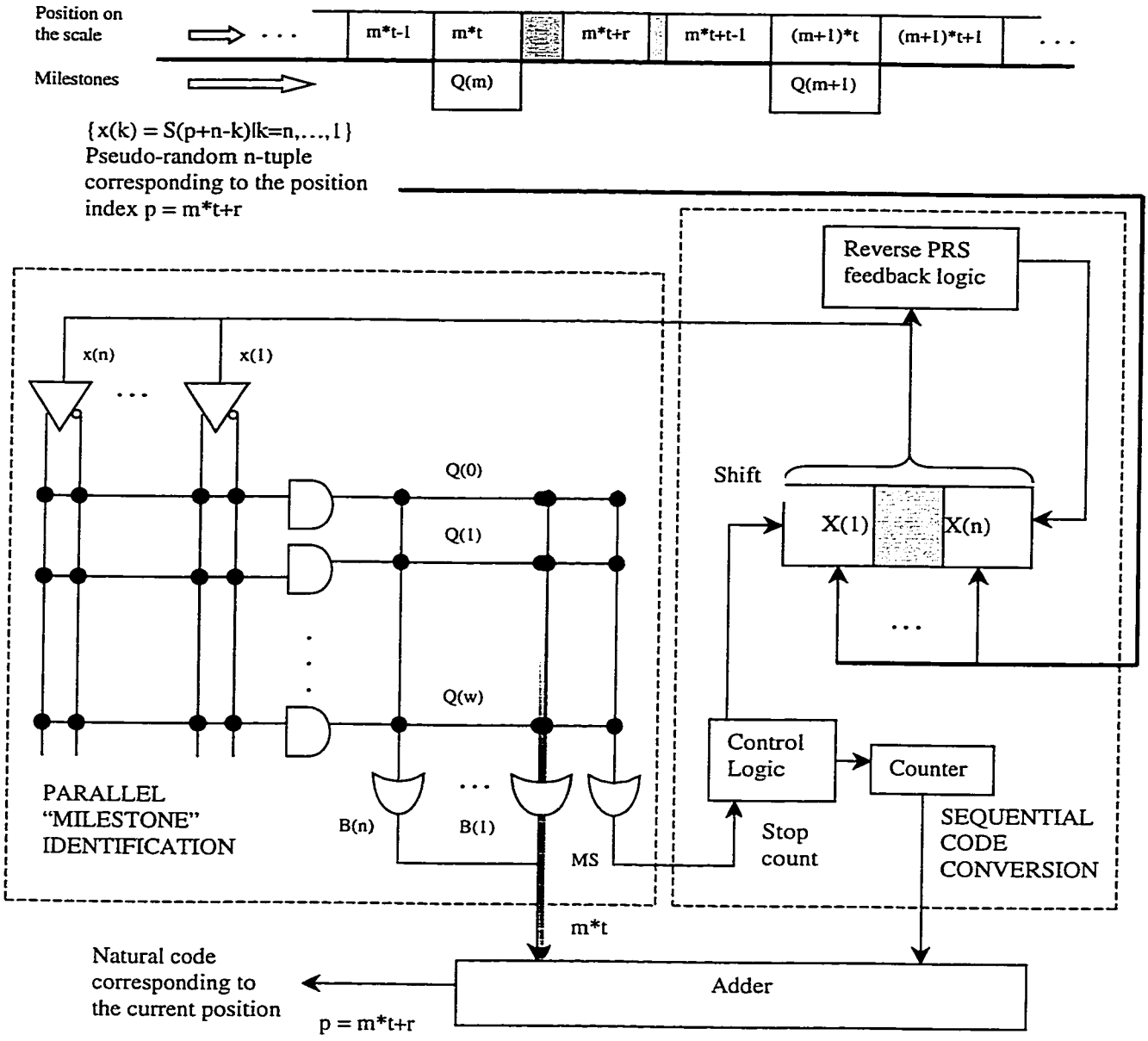


Figure 3.5 Serial/parallel pseudo-random/natural code conversion [41]

3.1.4 2D Pseudo-Random Window Index Recovery

The idea used for 2D pseudo-random window index recovery is similar to the 1D pseudo-random window index recovery [40].

Let us suppose that a and b are two PRSs defined by two primitive polynomials of degree m and n, over GF(p_1) and GF(p_2) respectively.

PRS1: $a_0, a_1, a_2, \ldots, a_u$

PRS2: $b_0, b_1, b_2, \ldots, b_v$

Let us consider A = {A1, A2, ..., Ap_1} and B = {B1, B2, ..., Bp_2} to be the sets of elements of GF(p_1) and GF(p_2). The following relations hold:

$$u = p_1^m - 1$$

$$v = p_2^n - 1$$

where:

u is the length of the PRS1 generated over GF(p_1);

v is the length of the PRS2 generated over GF(p_2);

A two dimensional pseudo-random encoded grid can be constructed by passing the PRS1 across the columns, and PRS2 across the rows.

For example, shown in Figure 3.6 is a 15 row-lines and 15 column-lines encoded with the terms of two pseudo-random-multi-valued-sequence "PRMVS" with $q = 2^2$ and m = 2, defined over GF(2^2) = (0, 1, w, w^2).

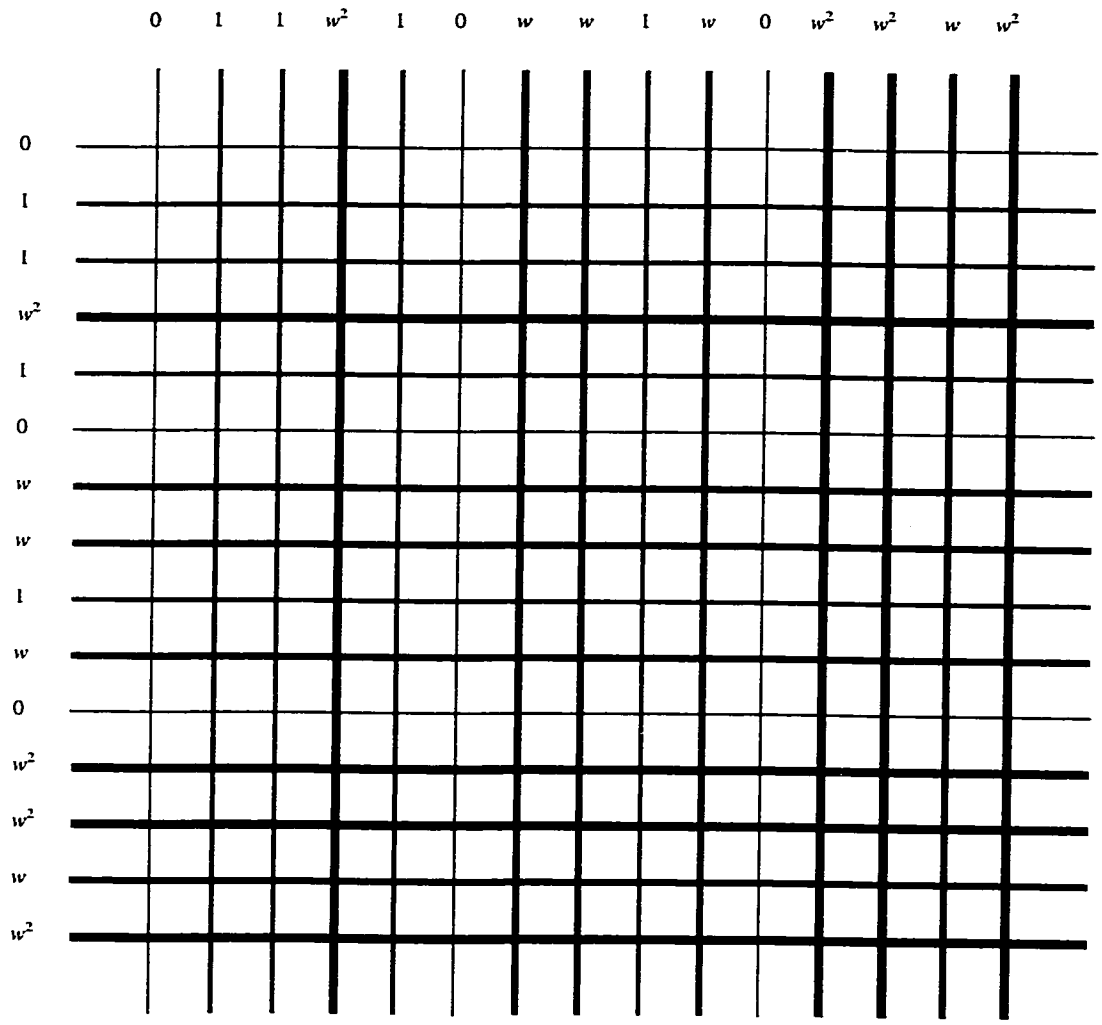


Figure 3.6 A 15-by-15 pseudo-random encoded grid

Any window of dimension mxm slid across that pseudo-random encoded grid is unique. The top left corner of the window W(i,j) is defined to be the origin of the window relative to the top left corner of the grid.

The problem of 2D window index recovery can be stated as follows:

> *Given the elements of an mxn-size window belonging to a pseudo-random encoded grid, determine the i and j indexes of the origin of the window W(i, j) relative to the top left corner of the PRG.*

Methods used for 2D window index recovery are the same as those used for 1D window index recovery (mentioned previously). The only difference is the recovery of the two indexes i and j of the origin of the window W(i,j) used in 2D PRG instead of one used in 1D PRS.

Chapter 4

3D Object Extraction and Reconstruction

Structured light is an efficient technique for obtaining 3D-scene information from a single 2D image by using a specially designed light source to project sheets or beams of light with a priori spatial distribution onto the scene casting lines or points onto the object. A camera is used to visualize from an angle the structured light projections on the object surfaces. The 3D-position recovery of the object can then be obtained using methods discussed in Chapter 2.

This chapter describes the method used in this thesis for visualizing/extracting 3D objects from 2D colored images, and its benefits over other methods used earlier (discussed in Chapter 2).

The following topics are arranged according to the order the program was designed and built for the object extraction from images.

4.1 Background Extraction

Extracting the background can be seen as detecting the foreground. The foreground corresponds to the object of interest. But before we proceed on explaining how the colored image background is extracted, some understanding of image types has to be explained.

4.1.1 Types of images

4.1.1.1 Binary Images

Binary images are the simplest type of images and can take on two values, typically black and white. A binary image is referred to as a 1-bit/pixel image because it takes only 1 binary digit to represent each pixel.

4.1.1.2 Gray-Scale Images

Gray-scale images are referred to as monochrome, or one-color, images. They contain brightness information only, no color information. The number of bits used for each pixel determines the number of different brightness levels available. The typical image contains 8 bits/pixel data, which allows us to have 256 (0-255) different brightness (gray) levels. Additionally, the 8-bit representation is typical due to the fact that the *byte*, which corresponds to 8-bits of data, is the standard small unit in the world of digital computers.

4.1.1.3 Color Images

Color images can be modeled as three-band monochrome image data, where each band of data corresponds to a different color. Typically color images are represented as red, green, and blue, or RGB images [56]. Using the 8-bit monochrome standard as a model, the corresponding color image would have 24 bits/pixel – 8 bits for each of the three color bands (red, green, and blue. Figure 4.1 illustrates a single pixel's red, green, and blue values that can be referred as a *color pixel vector* – (RGB).

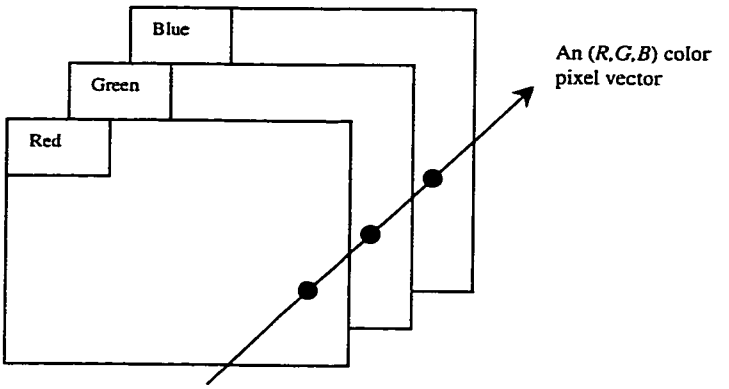


Figure 4.1 Color image composition

Having gained some understanding on images and their types, we can now proceed on explaining the method used for background extraction.

By capturing two images, one with only the background in it and the other with the background presented in it plus some extra feature using a still camera (the background does not change), the background can then be extracted and the foreground detected simply by subtracting these two images from each other. In an ideal case, subtracting two images with the same background, results in an image with a black background – background detected (i.e. two identical pixels subtracted from each other results in a zero pixel – black pixel). But in normal cases, specially when dealing with colored images, this simple subtraction will not work properly and a lot of noise will be introduced to the resulting image. The reason is that there are fluctuations caused by natural phenomena that add a random value to the exact brightness value for a given pixel [23]. Subtracting a image pixel from that same pixel in another image with extra brightness introduced, will not produce a zero pixel. This problem was solved in this thesis through the use of hue/saturation/lightness color transform.

4.1.1.3.1 Hue/Saturation/Lightness "HSL"

The Hue/Saturation/Lightness color transform allows us to describe colors in terms that can be more easily understood (see Figure 4.2). The *lightness* is the brightness of the color. Given the RGB color values of a specific pixel, its lightness can be calculated as follows:

$$Lightness = R + G + B;$$

The *hue* is what normally we think of as "color" (for example, green, blue, yellow, etc.) Its value ranges between 0 and 360 (see Figure 4.2). It can be defined using RGB by the following program fragment:

$$Hue = \cos^{-1}\left\{\frac{\{1/2[(R-G)+(R-B)]\}}{\sqrt{(R-G)^2+(R-B)(G-B)}}\right\}$$

$$\text{If } B > G \text{ then } Hue = 2pi - Hue; \qquad [27]$$

The saturation is a measure of how much white is in the color. For example, pink is red with more white, so it is less saturated than a pure red. Its value ranges between 0 and 100 as shown in Figure 4.2. It can be calculated using the following equation:

$$\text{Saturation} = 1 - \frac{3\min(R,G,B)}{Lightness};$$

Most people can relate to this method of describing colors. For example, "a deep, bright orange" would have a large Lightness ("bright"), a hue of "orange", and a high value of saturation ("deep"). We can picture this color in our minds, but if we defined this color in terms of its RGB components, R = 245, G = 110, and B = 20, most people would have no idea how this color appears. Figure 4.3 shows all possible colors based on Hue and Saturation values.

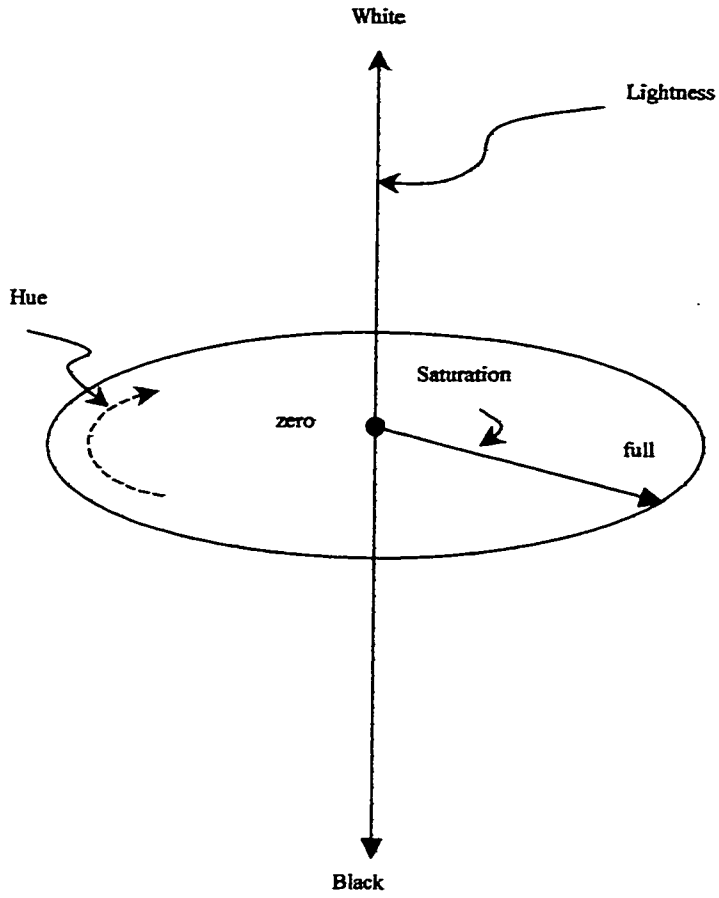


Figure 4.2 HSL color space

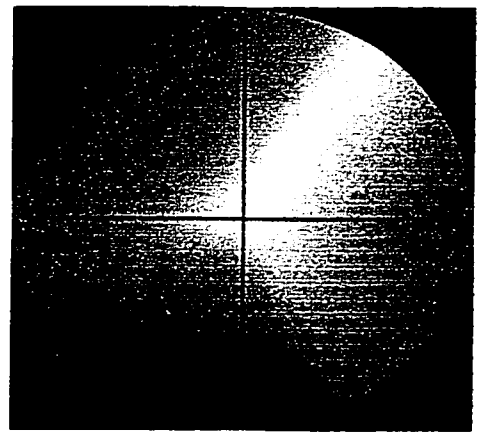


Figure 4.3 Hue and Saturation color arrangement [42]

Knowing that it is possible to split each color into three separate values Hue/Saturation/Lightness, the problem explained previously in extracting the background due to fluctuations in brightness (lightness) values in pixels can now be resolved. By isolating the brightness and calculating the mean and variance of the hue (μ_h, σ_h) and saturation (μ_s, σ_s) of every pixel in the first image and comparing those values with the same pixel coordinates in the second image, noise generated by brightness fluctuations is eliminated and the background can be easily detected.

4.2 Real-time Grid color selection

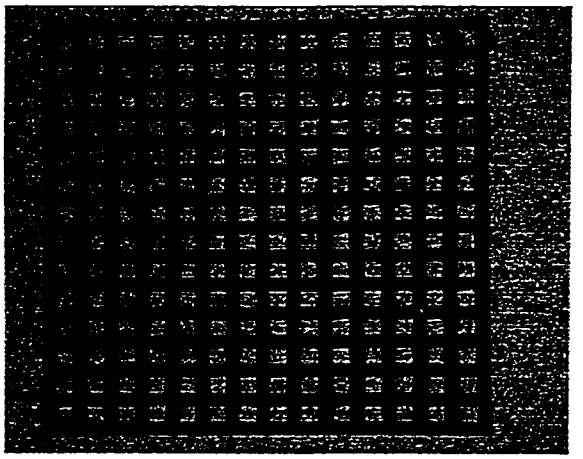
In order to detect the grid being captured by a camera after being projected on a natural scene/environment, careful selection of this color has to be wisely chosen. Having a grey background for instance, projecting a light blue grid on that background won't be a good idea, since it might be assumed of being part of that background and won't get detected. An approach proposed in this thesis is done by projecting different grid colors on a scene one by one, without any priori knowledge of that scene, and trying to threshold each one of them after the background has been eliminated. The success rate (calculated and displayed using a histogram) of that extracted grid determines how well the color is to be used in this environment.

4.2.1 Thresholding

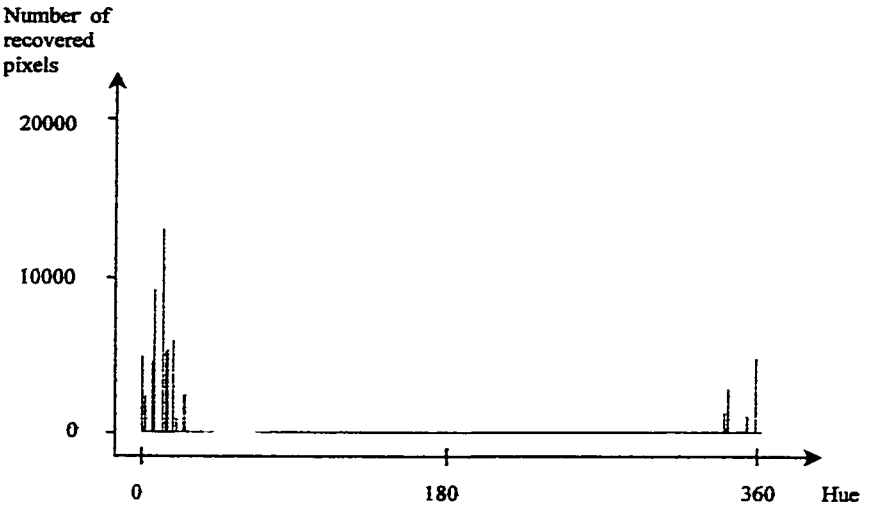
Image thresholding is a technique for converting a grey-level image into a black and white image. Pixel values below a specified threshold T are converted to black, whereas pixel values at or above the threshold T are converted to white.

4.2.1.1 Masked threshold

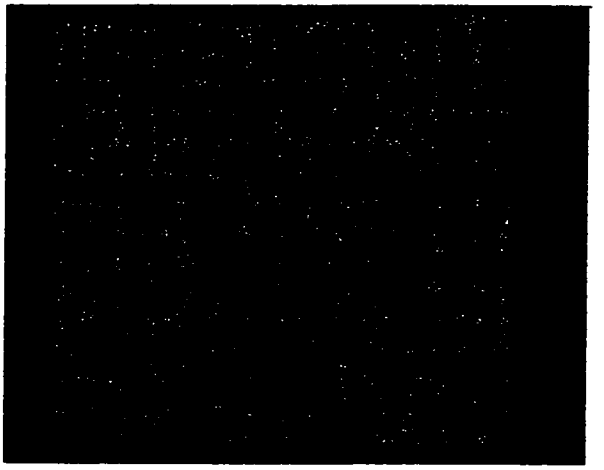
Sometimes it is desirable to compute the threshold value for only a certain part of the object. In this case, a mask is used to threshold only that certain grid color after being projected onto the scene.



(a) Projected grid lines on a flat surface



(b) Histogram of the detected grid



(c) Grid recovered after thresholding

Figure 4.4 Recovered grid from grid line projections

The histogram is normally generated by looking at all the pixels in the image. With a mask, the histogram is computed only for the pixels that are defined in the mask.

Figure 4.4(a) displays the grid line projections on a flat surface. In Figure 4.4(b), the histogram of all the recovered pixels of the grid line projected is displayed after applying a masked threshold to that color. Figure 4.4(c) shows the grid obtained after removing the background and displaying it through the recovered pixels.

4.3 Pseudo-Random Multi-Valued Sequence "PRMVS" Encoding

As has been discussed in chapter 2, there has been many methods proposed for resolving the *point identification* problem - identifying coordinates of projected dots in an image (Figure 4.5). Projecting dot (line) by dot (line), and identifying each one of them before the other is projected is one way but is time consuming. Changing the color, thickness, etc. for each of the lines is another way, which also has limits and requires a priori knowledge of the lines and there position with respect to the other lines before being projected.

This thesis discusses the efficiency of the pseudo-random colored sequence encoding for solving the point identification problem in structured light applications. As mentioned in the previous section, careful selection of grid colors has to be considered, since this method should work in any colored scene/environment. Through the use of pseudo-random colored sequences, the priori knowledge of line colors and their position with respect to the others is eliminated. At first, a pseudo-random binary color encoded sequence "PRBCS" grid, with a Galois field GF(2), q=2, m=4, and $h(x) = x^4 + x + 1$, was projected onto the scene after selecting the best two colors for that specific scene. Any 4x4 window slid over that mesh is unique with a position that can be identified easily (see Figure 4.6). The only problem in using PRBCS on a 15x15 projected mesh is its large window size used for the point identification. Any loss of information (i.e. disconnected

lines, missing corners, etc) in the extracted grid will result in the lack of sufficient information required by the window in identifying the points in that area.

An alternative for solving that problem is by either increasing the resolution of the projected grid (increasing the number of row and column grid lines), or by using pseudo-random mutli-valued color encoded sequence "PRMVCS" grid, which will result in a smaller window size and hence reducing the amount of information required for point identification.

This thesis was later upgraded to PRMVS using four colors. By projecting different colors onto the scene, the best four colors were selected. A 15x15 PRMVS grid having a Galois field $GF(2^2)$, $q= 2^2$, $m =2$ and $h(x) = x^2 + x + w$ was then projected on the scene using those four selected colors. By doing this, only a 2x2-window size was needed to extract the same information covered by 4x4 window on a 15x15 PRBS grid (see Figure 4.7). Besides, by decreasing the window size, less area in the 15x15 mesh is covered and hence an increase in the amount of information (extracted coordinates).

By using the best four colors though, the extracted grid will not be as good as when using only the best two colors, which will result in turn in more information loss that is required for reconstructing the grid (discussed in section 4.5). This problem was resolved by first projecting the PRMVS grid on the scene for the identification of colors of all the grid columns and rows. The best unary color was then projected onto the same scene for extracting the best grid. The final grid used for point identification was later reconstructed using the unary colored extracted grid and its column and row colors were then chosen based on the PRMVS extracted grid.

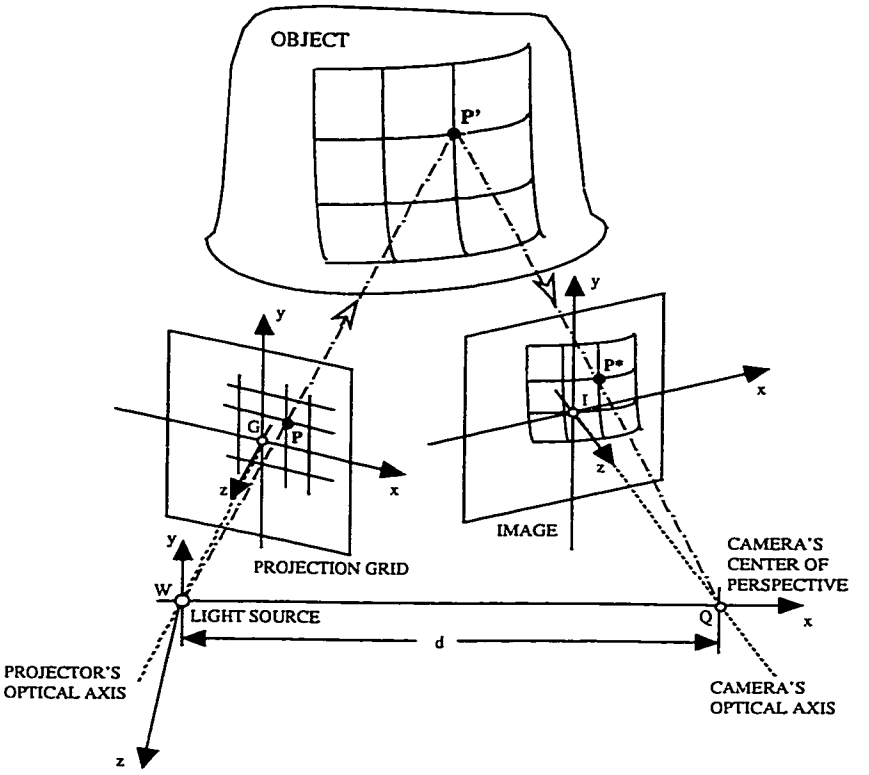
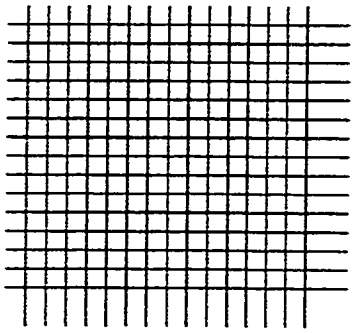
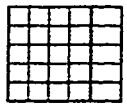


Figure 4.5 "*Point identification*" in structured light

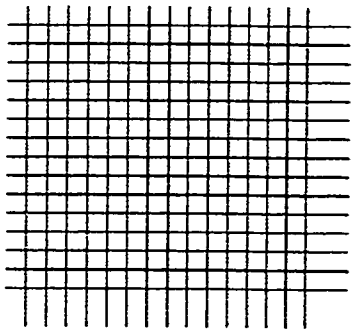


(a) A 15x15 PRBCS encoded mesh



(b) A 4x4 unique window obtained from the encoded mesh

Figure 4.6 Pseudo-random binary color encoded mesh



(a) A 15x15 PRMVCS encoded mesh

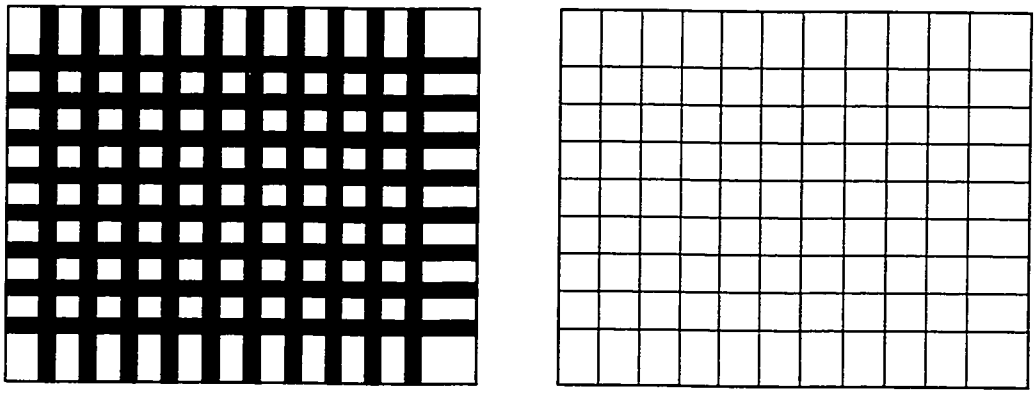
(b) A 2x2 unique window obtained from the encoded mesh

Figure 4.7 Pseudo-random multi-valued color encoded mesh

4.4 Grid Line Edge Detection

By applying pseudo-random encoding to structured light grid, the point identification problem is resolved. But before extracting those captured pixel coordinates generated from the intersection of projected grid lines, a method has to be applied to reduce the error and increase the possibility of those mapped pixel coordinates to the projected grid line intersections obtained through triangulation.

A method proposed by Hu et al. [4] where a thinning algorithm is applied to the captured grid stripes to obtain the skeleton and reduce the stripes to one pixel wide (see Figure 4.8). By doing this, mapping between the intersecting stripes and the intersecting projected lines is reduced to one pixel. But in this case, some accuracy is lost, since that pixel selection depends mainly on the thinning algorithm [52] [53] [54], and not on the best choice mapping between the intersecting projected grid lines, and the captured intersecting stripes.



(a) Captured Grid stripes before thinning

(b) Captured Grid stripes after thinning

Figure 4.8 Grid shape before/after thinning

A new method proposed in this thesis, through the use of grid strip edge detection.

Edge detection is used to find object boundaries by marking potential edge points corresponding to places in an image where rapid changes in brightness occur. After these edge points have been marked, they can be merged to form lines and object outlines.

There are several edge detection operators. They are based on the idea that edge information in an image is found by looking at the relationship a pixel has with its neighbors. If a pixel's gray-level value is similar to those around it, there is probably not an edge at that point. However, if a pixel has neighbors with widely varying gray levels, it may represent an edge point. In other words, an edge is defined by a discontinuity in gray-level values. Ideally, an edge separates two distinct objects. In practice, apparent edges are caused by changes in color, texture, or by the specific lighting conditions present during the image acquisition process.

4.4.1 Sobel Operator

The Sobel edge detection [23] [55] masks $\mathbf{H_1}$ (row mask) and $\mathbf{H_2}$ (column mask) used, measure the gradient of the image *u(m,n)* in two orthogonal directions (see Figure 4.9). The masks are as follows:

$$H_1 = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \qquad H_2 = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

These masks are each convolved with the image. In each pixel location there are two values now: $s_1(m,n)$ corresponding to the result from the row mask, and $s_2(m,n)$ from the column mask. These values obtained are used to calculate the edge magnitude and direction as follows:

$$s(m,n) = \sqrt{s_1^{\,2}(m,n) + s_2^{\,2}(m,n)}$$

$$\theta_s(m,n) = \tan^{-1}\left[\frac{s_1(m,n)}{s_2(m,n)}\right]$$

The pixel location *(m,n)* is declared an edge location if *s(m,n)* exceeds some threshold *t*. The locations of edge points constitute an edge map $\varepsilon(m,n)$, which is defined as:

$$\varepsilon(m,n) = \begin{cases} 1, & (m,n) \in I_s \\ 0, & otherwise \end{cases}$$

where

$$I_s = \{(m,n); s(m,n) > t\}$$

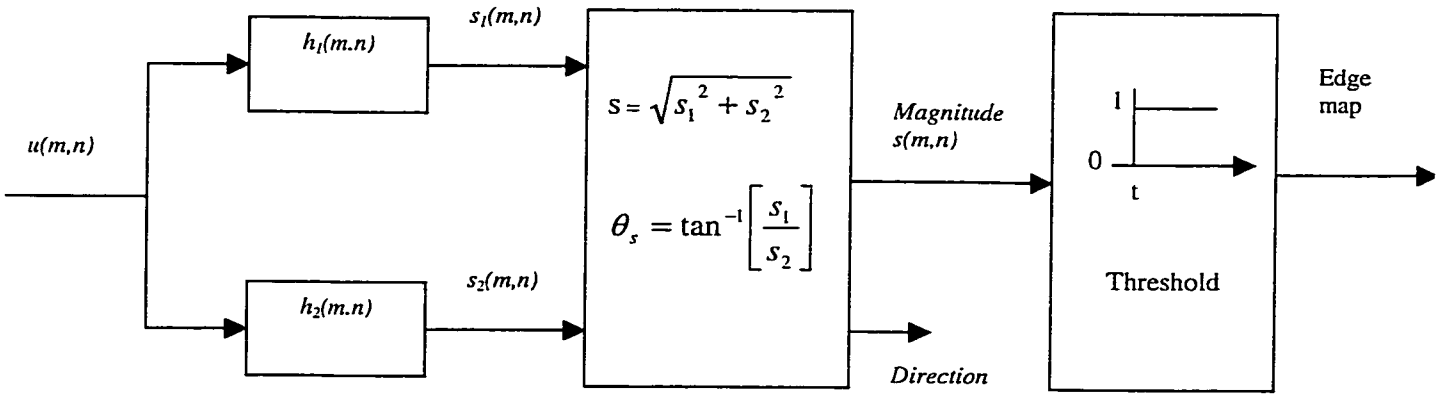
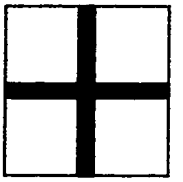
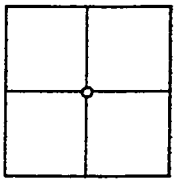


Figure 4.9 Edge detection via Sobel operator

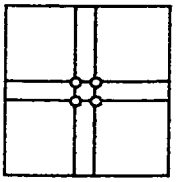
Recovery of the edges of the grid will yield the four corners at the intersection of two stripes, whereas by obtaining the skeleton will yield only one point. This new approach results in a 4:1 increase in the density of the points used for triangulation as shown in Figure 4.10.



(a) Intersecting grid strips

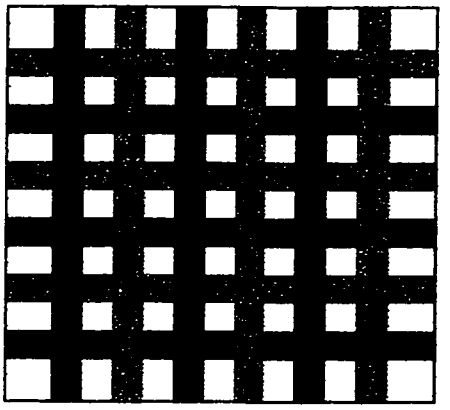


(b) Grid strips after thinning

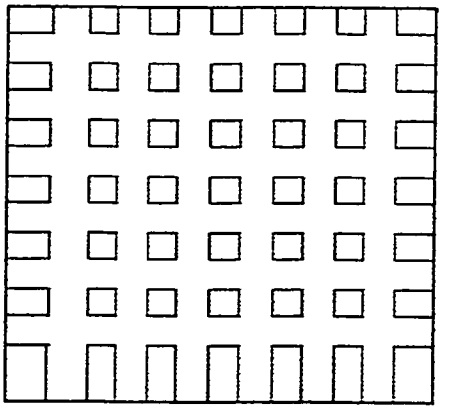


(c) Reconstructed grid strips after edge detection

Figure 4.10 Recovered grid from thinning or edge detection



(a) Extracted grid stripes



(b) Stripes obtain after edge detection with their corresponding colors

Figure 4.11 Grid color stripes before/after edge detection

Moreover, using edge detection increases the mapping probability between the corners obtained from the intersecting lines and those of the extracted grid stripes. A disadvantage though is the need for reconstruction of the grid after edge detection (explained in the next section), that can be automatically obtained when finding the skeleton of the projected grid. Figure 4.11 shows an extracted grid with arbitrary selected stripes and the recovered stripes after edge detection.

Another very important disadvantage that needs to be considered in the use of edge detection instead of just finding the skeleton of the projected grid, is the corruption of the pseudo-random encoding applied to the projected grid. For example, having a projected grid with 15-bit PRBS 000111101011001 for both rows and columns, every 4x4 window slide over that grid is unique. Finding the skeleton of that grid will preserve that encoding sequence. But finding the edges of the pseudo-random encoded stripes and rebuilding the grid will generated the following sequence 000000111111110011001111000011, that is two lines of the same color for every projected stripe, which is not correct. A way to overcome that problem is by labeling the corners for all connected edges, shown in Figure 4.11(b), before grid reconstruction: UPPER_LEFT, UPPER_RIGHT, LOWER_LEFT, LOWER_RIGHT. By considering the lines that only pass through the "UPPER_LEFT and LOWER_LEFT" or "UPPER_RIGHT and LOWER_RIGHT" corners for columns, and the lines that pass through the "UPPER_LEFT and UPPER_RIGHT" or "LOWER_LEFT and LOWER_RIGHT" corners for rows, the encoding sequence of the projected grid can be maintained. Different grid lines are selected based on the coordinates that need to be calculated.

4.5 Grid Reconstruction

After obtaining the edges from the grid stripes, a problem is generated in connecting those recovered edges to obtain a new grid that matches the exact color specifications of the old extracted grid stripes. For example, a grid shown in Figure 4.12 has to be reconstructed from the extracted edges in Figure 4.11(b). The problem is not as difficult as it may seem in an ideal case. Having recovered the edges of a pseudo-random colored

encoded grid and knowing that a unique window can be slid over that encoded grid, then connecting those edges is only a matter of matching them with their neighbors in that specified window.

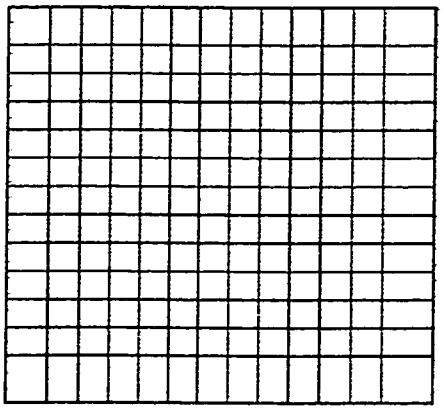


Figure 4.12 Rebuild new grid from recovered edges

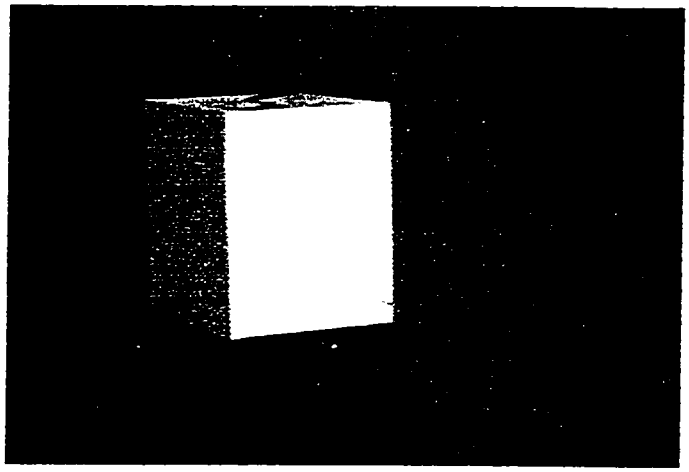
But in normal cases, connecting those edges is not a simple matter. For instance, a presence of a 3D object in the scene might cause the extracted grid stripes to be bent, shifted and/or even broken. This kind of shift in lines might fool the program in making incorrect decisions. Noise also plays an important role in the generation of error. The following considerations have been followed which will reduce or eliminate the error generated in connecting the stripes and building the grid.

(a) Connection of edges that only agree with the pseudo-random encoding setup used.
(b) Removal of any loose lines not part of a square.
(c) Deletion of any points which could not be identified of being part of the grid.

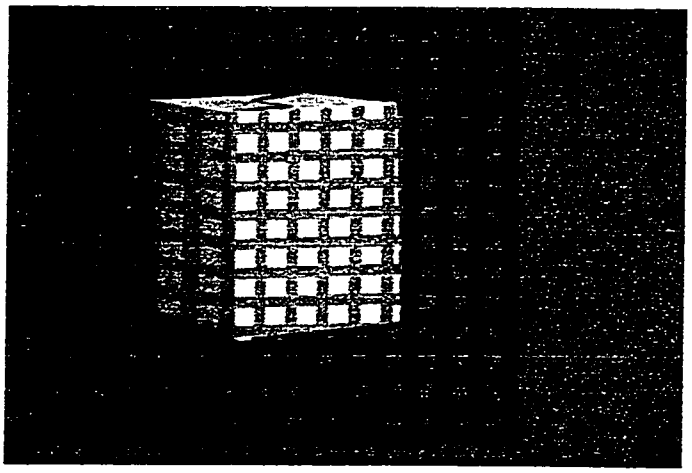
After the construction of that new grid, all intersection points between stripes are scanned and saved for later use in identifying the 3D object from the scene.

4.6 Detecting 3D Objects

A 3D object is introduced to the scene, and all the previous steps discussed earlier are applied step-by-step. The catch here is that when a structured light is projected onto a 3D object, the projected lines will get deformed. This deformation is directly proportional to the object's shape, orientation and displacement from the camera, as shown in Figure 4.13. By just looking at those deformations, one is able to get some idea on the object's shape. Moreover, depth measurements are dependent on the light projector from the camera. The larger the displacement, the more accurate depth measurements can be made, as discussed in chapter 2. Grid built after edge detection and connection of the grid stripes will also be deformed, since the original extracted grid from the scene is deformed. Scanning and comparing all the stripe intersecting points of the grid before and after deformation through the use of PRS encoding, an estimate depth calculation of the object in the scene can be attained. Hence 3D object shape can be obtained from the grid line deformations and the 3D object depth by comparing of the intersection points in the two grids with/without 3D object in the scene and calculating the difference in the shift of each point in those two grids [35]. A summary of the steps involved for calculating the shape and depth of a 3D object in the scene is shown in Figure 4.14.



(a) Introduction of an object to the scene



(b) Projected structured light and grid deformation

Figure 4.13 Grid deformation on presence of 3D object in the scene

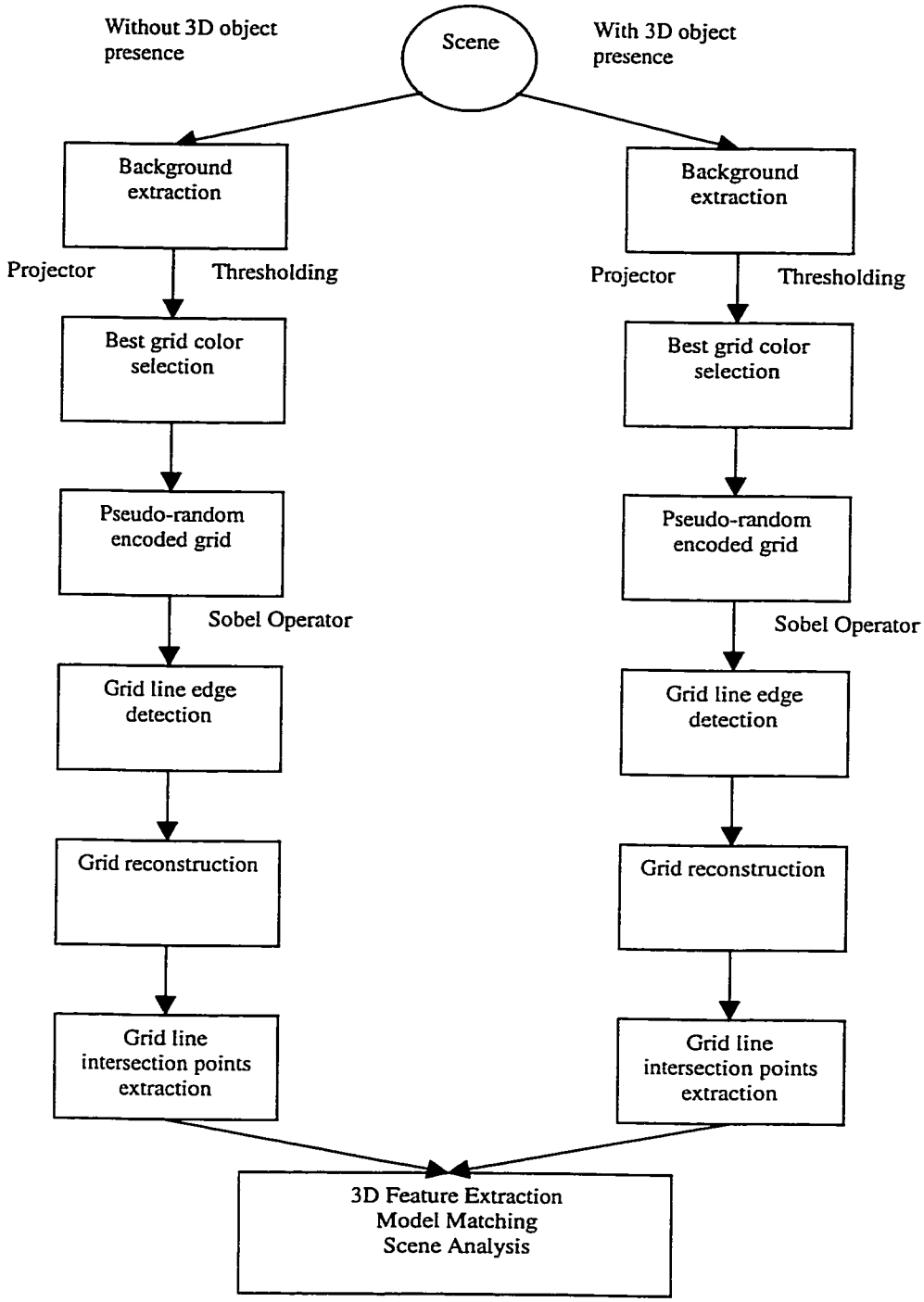


Figure 4.14 System Diagram

Chapter 5

Experimental Results

In the previous chapter the steps required to extract 3D objects from images were presented. In this chapter, the Graphical User Interfaces (GUI) are shown and different experimental test cases are presented to validate the developed software and the image processing algorithms.

The experimental set used in this thesis (Figure 5.1) consists of an Epson ELP-3000 projector, 2 Pentium computers and a Sony CCD-FX310 camcorder. The projector projects the grid, presented by one Pentium computer, on the scene. The camcorder captures the grid and feeds it directly to the other Pentium computer, which does the rest of the processing for the 3D-object extraction.

All programs for this thesis were developed using Visual C++ 6.0 and the MFC, with the aid of Microsoft Vision SDK 1.1 for image processing purposes.

5.1 Graphical User Interface (GUI)

The GUI shown in Figure 5.2 is the one used by one of the Pentium machines to select the best grid and project it over the scene. The "First Color", "Second Color", "Third Color" and "Fourth Color" are pull down menus used to select the colors for the PRMVS encoded grid. In Figure 5.2 for example, the "First Color" is chosen as red, the "Second Color" as yellow, the "Third Color" as magenta, and the "Fourth Color" as green. There are 16 different colors to choose from in each menu. "Rows" and "Columns are also menus used to choose the number of lines for each of the grid rows and columns respectively. One is able to choose from a selection of 15, 31, 63, 127 and 255 lines. The grid is pseudo-random encoded automatically based on the number of line selection for both rows and columns. In this thesis a 15x15-line grid is used as shown in Figure 5.2.

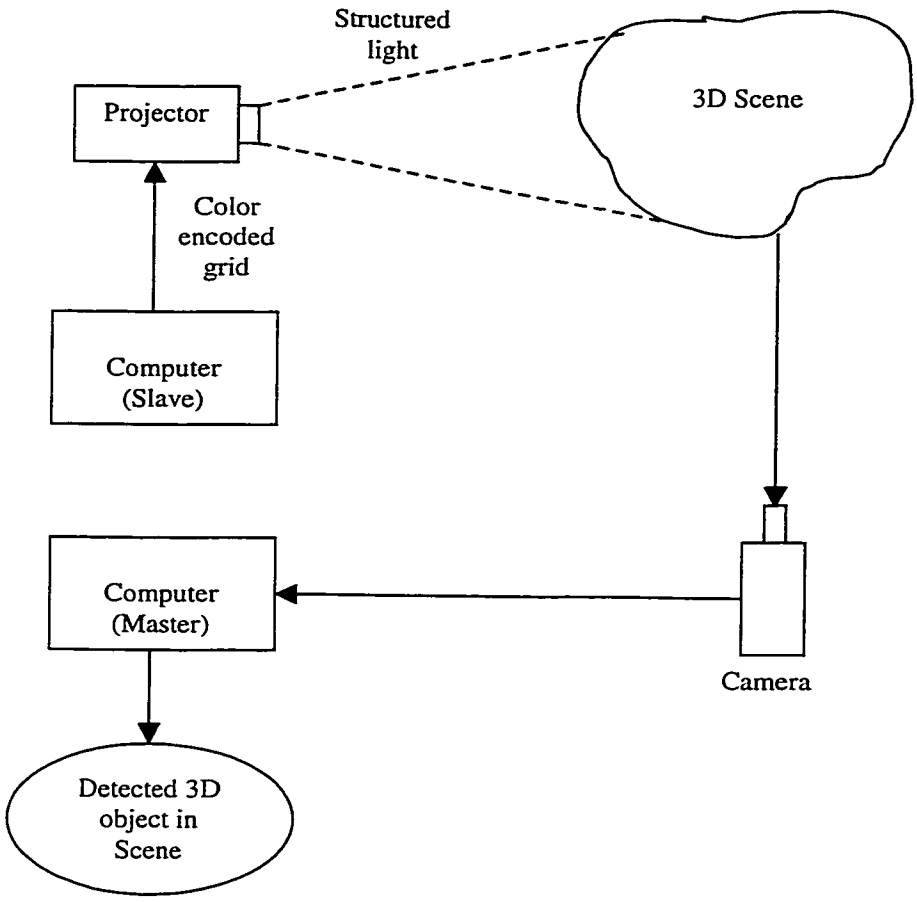


Figure 5.1 Experimental set

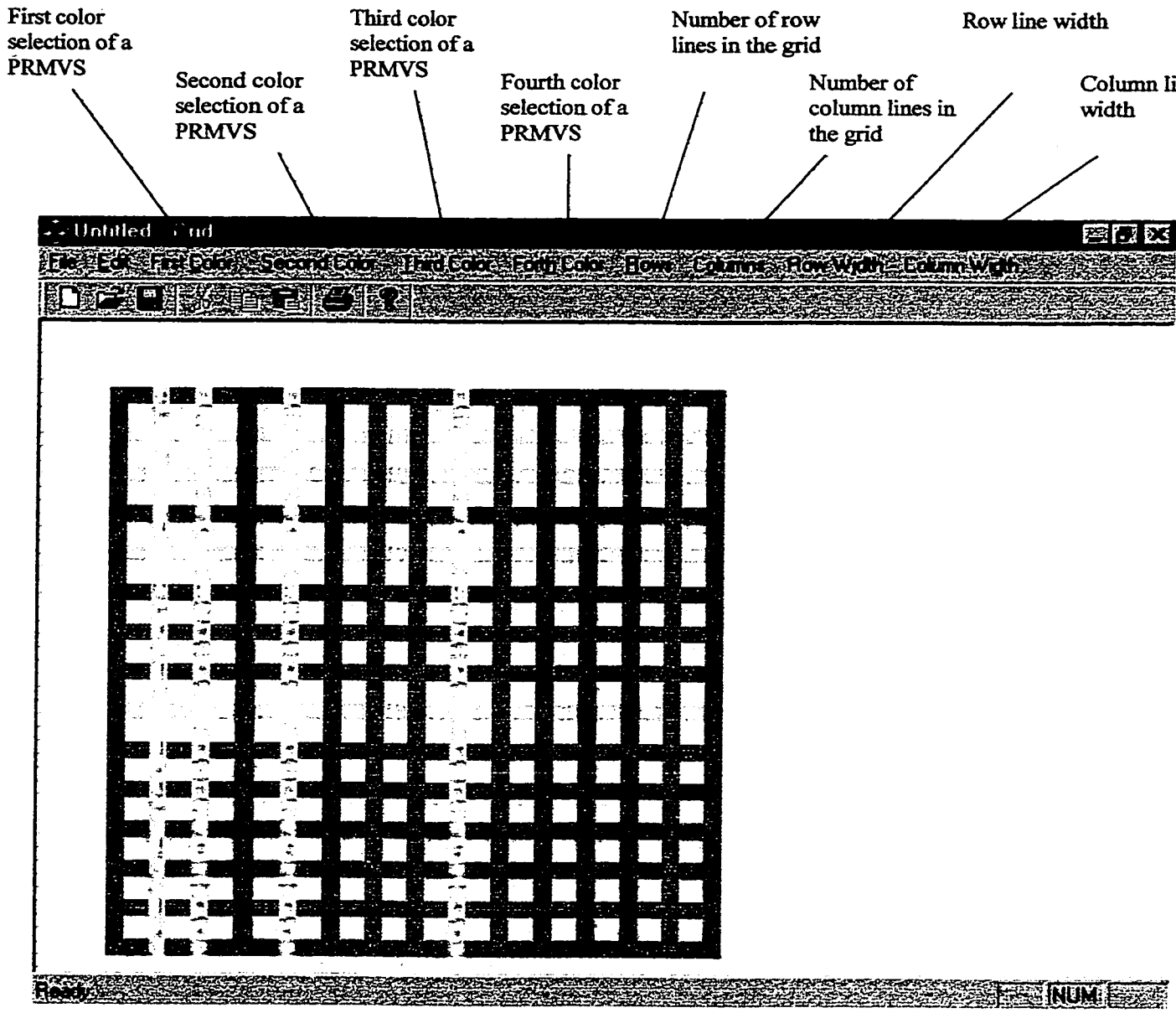


Figure 5.2 Grid selection interface

"Row Width" and "Column Width" are used to select the line width for the row and column lines respectively.

Figure 5.3 displays the GUI used on the other Pentium computer for processing the images captured by the camera. The following explains all the button functionality found on that GUI in order from left to right.

The first is the "D" button used to select the image source provider that supports video for windows standard. Next comes "C" button which is used to select the frame compression required (i.e, Microsoft MPEG-4 Codec, Microsoft Video1, Cinepak Codec by radius, Full Frames (uncompressed)). The default compression rate was selected using the Microsoft Video1. The "F" button is used for the video capture format settings. The "RGBT: 24bit RGB" was selected. The "S" button is used to display a dialog for video source settings. The dialog allows to choose the image source provider (camera, VCR, etc.), the standard (NTSC, PAL) and the picture adjustment parameters. The image size preferred can be selected from the "Image Size" pull down menu. In order to update the view whenever a new image is captured the "U" button must be pressed. The "ONE" button is used to display the captured image before/after processing depending on the other buttons selected. In order to extract the grid from the image the "SUB" button must be used. By pressing that button the background is removed leaving the image with nothing but the projected grid. The "HIST" button is used to display a histogram of the projected grid color. By changing the projected grid color and trying to extract that color from the image, one is able to select the best color of the grid for future processing. The value displayed in the status bar also aids in the selection of the best grid. This value displays the best grid color extracted from the image. The "F" button is used to select the image with the projected grid on, before introducing an object into the scene. The "S" button is used to select the image with the projected grid on, after introducing an object into the scene. Those last two buttons are used to compare the two images before/after the introduction of an object to the scene for object depth calculations.

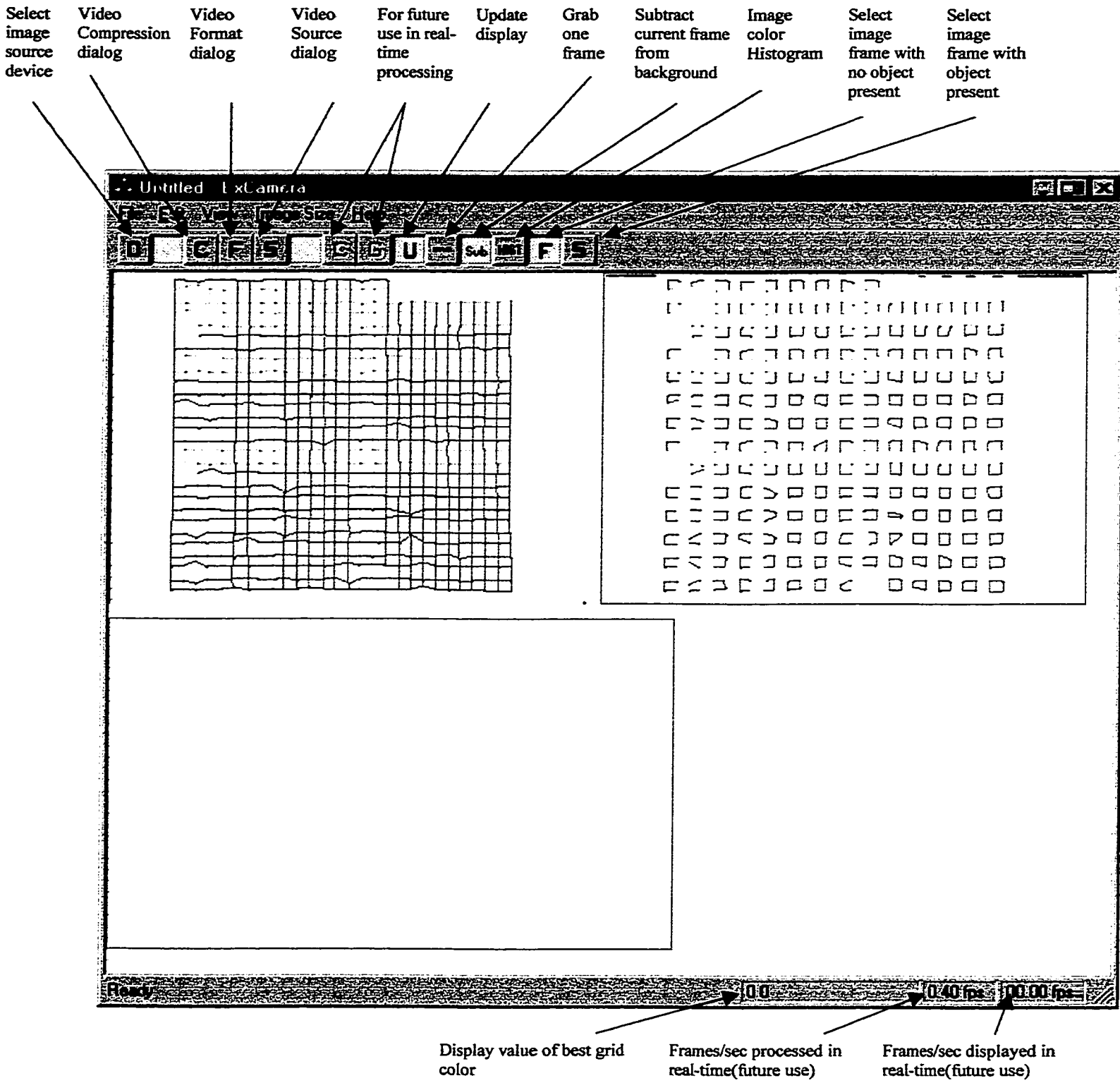


Figure 5.3 3D object detection/extraction interface

5.2 *Test Cases*

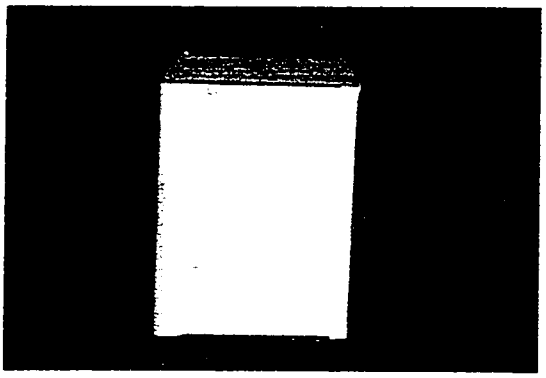
The following are some test cases used in this thesis for extracting 3D object shapes. In the first test case, the effect of different colored scenes/environments on the extraction of projected colored grids is shown and explained for careful grid color selection. In the second test case, some examples are used to demonstrate the success of this project in extracting 3D objects from a scene and calculation of their absolute depth.

5.2.1 *Test Case I*

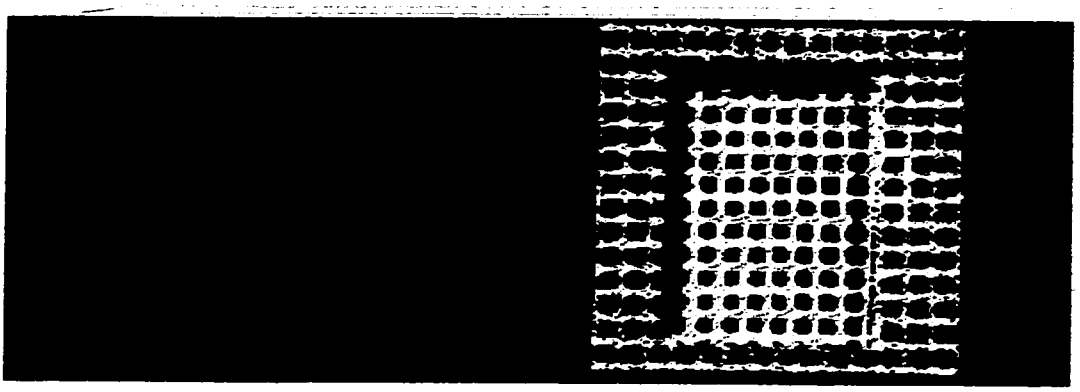
Knowing that a 3D object has to be detected and extracted in any colored scene/environment, careful grid color selection has to be considered (see chapter 4, section 4.2 for more details).

Looking at Figure 5.4 a), a white box is present in front of a dark gray background. In order to extract that object (box) and its coordinates, a number of different colored grids are projected onto the scene. The recovered grids, which are least affected by the scene after the background removal are the best grid colors. Projecting a blue grid on the scene and trying to extract it, produces the image shown in Figure 5.4 b). Only 20994 blue grid pixels were recovered. Whereas projecting a yellow grid produces 27848 yellow grid pixel (see Figure 5.4 c)). A much better result is shown in Figure 5.4 d) after projecting a green grid onto the scene, 32691 green grid pixels are recovered. Projecting a green grid on the scene, much of the grid is recovered after the background removal. By projecting a red grid, most of the grid is recovered with minimum amount of information lost 36175 red grid pixels recovered (see Figure 5.4 e)).

Careful color selection has to be considered, which is least affected by the scene in order to extract the maximum amount of information about the object's shape being considered and its coordinates.



a) Captured Image



b) Extracted blue grid

c) Extracted yellow grid

d) Extracted green grid

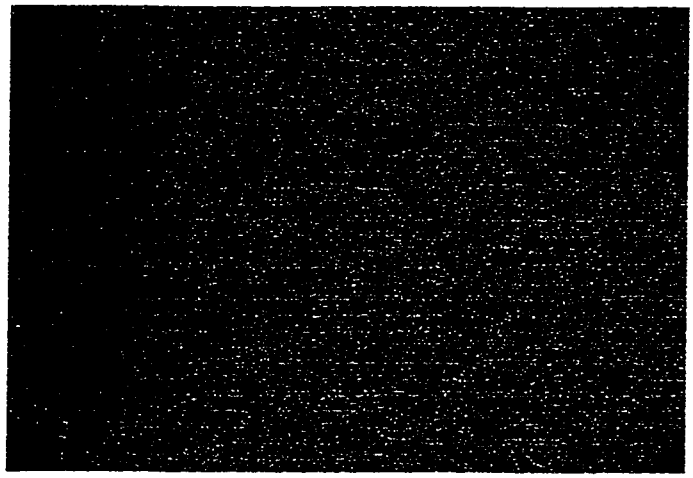
e) Extracted red grid

Figure 5.4 Recovered grid based on the projected color

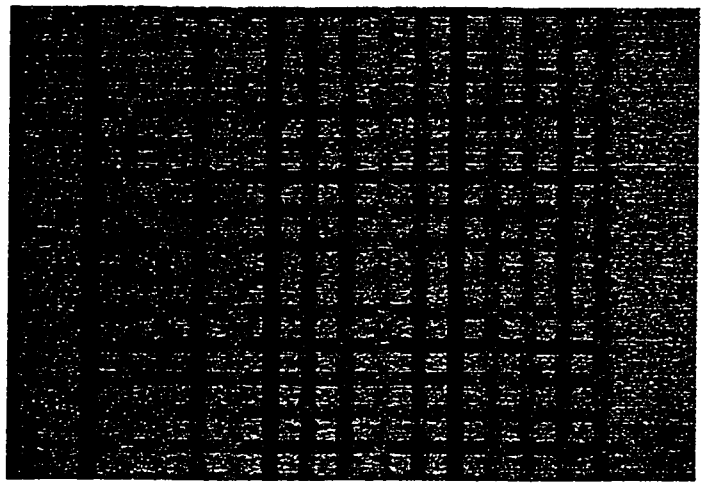
5.2.2 Test Case II

Having selected the best four grid colors for a specific scene, the 15x15 PRMVS encoded grid, with Galois field $GF(2^2)$, $q = 2^2$, $m = 2$ and $h(x) = x^2 + x + w$ for both rows and columns, is projected onto that scene (see Figure 5.5 b), Figure 5.6 b) and Figure 5.8 b)). The unary color (red was best suited of that specific scene/background) is then projected onto the scene, as shown in Figure 5.5 c), Figure 5.6 c) and Figure 5.8 c), for extracting the best grid with lease amount of noise and maximum amount of information (refer to chapter 4 section 3 for more details). When the grid is projected onto a scene with a 3D object in it, as shown in Figure 5.6 b) and Figure 5.8 b), the grid is uniquely deformed according to the object's shape and position from the camera. This deformation is proportional to the object's depth from the camera. By extracting the projected grid edges before/after the introduction of the 3D object (Figure 5.5 d), Figure 5.6 d) and Figure 5.8 d)) and trying to reconstruct the grids (refer to Figure 5.5 e), Figure 5.6 e) and Figure 5.8 e)) using those edges, all pixel points of the intersecting grid lines are unique and can be identified (PRMVS grid). The depth of that object from the camera is calculated by obtaining those grid line intersection points from the reconstructed grid before the introduction of the object and comparing them to their corresponding points obtained after the introduction of the object. The 3D extracted data used in reconstructing the 3D shape of both the cube and cylinder can be found in appendix C. Figure 5.7 and Figure 5.9 reveals the 3D shape of the extracted cube and cylinder respectively.

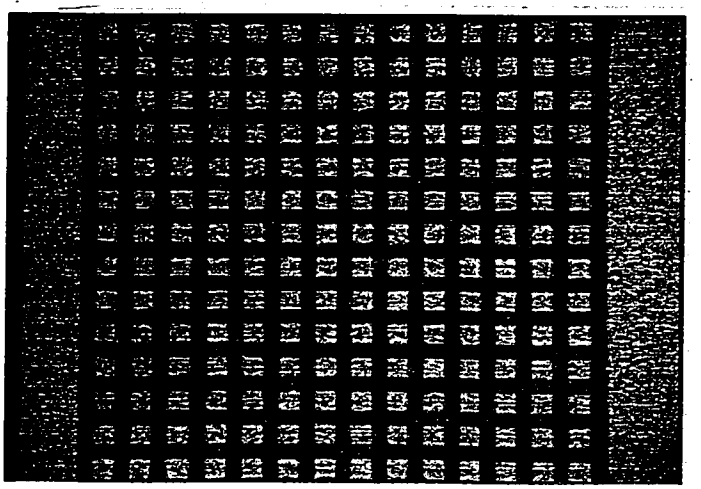
These experiments have validated the practical applicability of using PRMVS encoding and projected grids combined for 3D object detection and extraction in any colored scene.



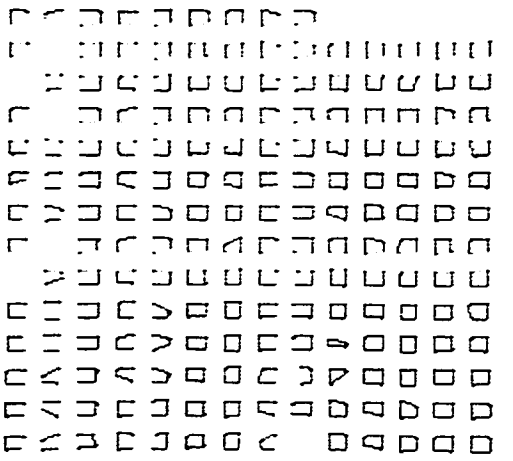
a) Background



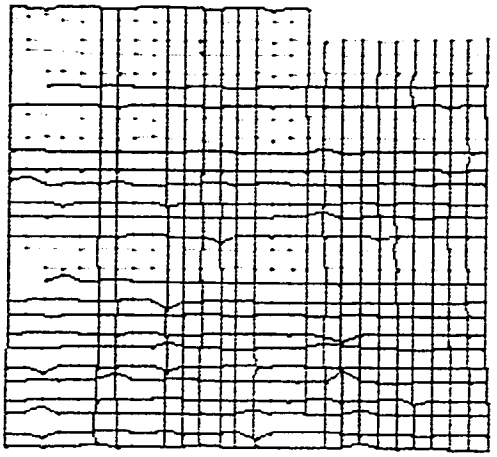
b) Projection of PRMVS grid onto the scene



c) Projection of uni-colored grid onto the scene

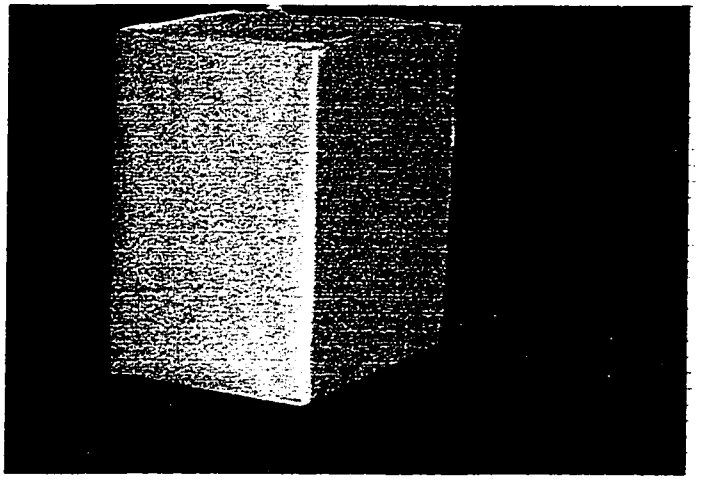


d) Extraction of grid edges

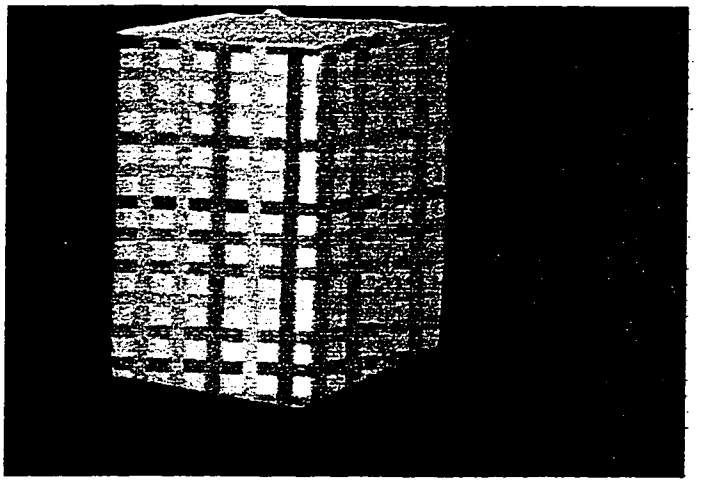


e) Reconstruction of the grid

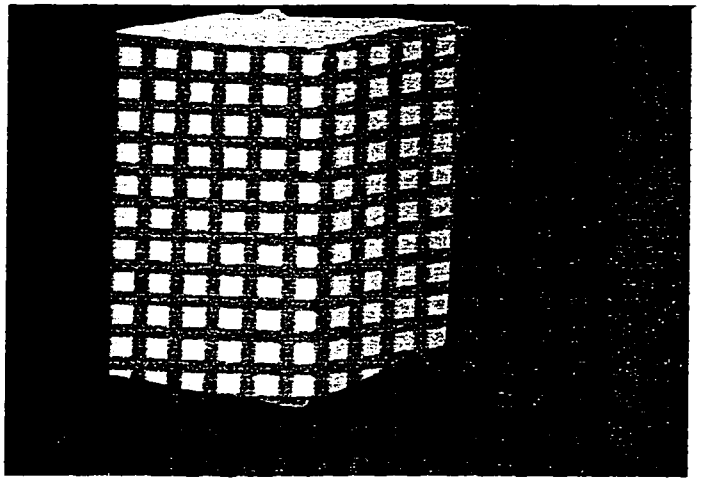
Figure 5.5 Background grid extraction/reconstruction



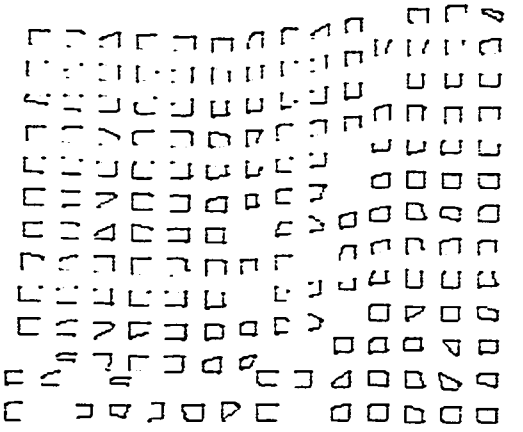
a) Introduction of cube onto the scene



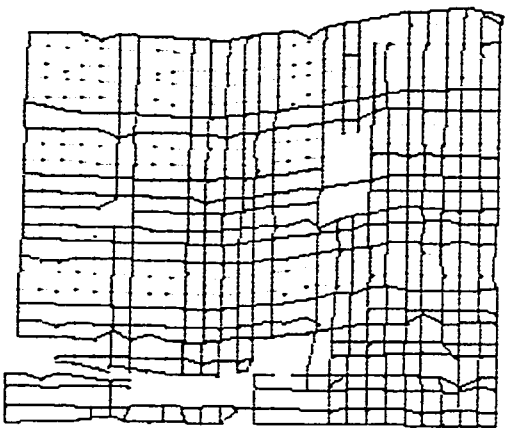
b) Projection of PRMVS grid onto the scene



c) Projection of uni-colored grid onto the scene



d) Extraction of grid edges



e) Reconstruction of the mesh

Figure 5.6 Grid extraction/reconstruction (with cube present in the scene)

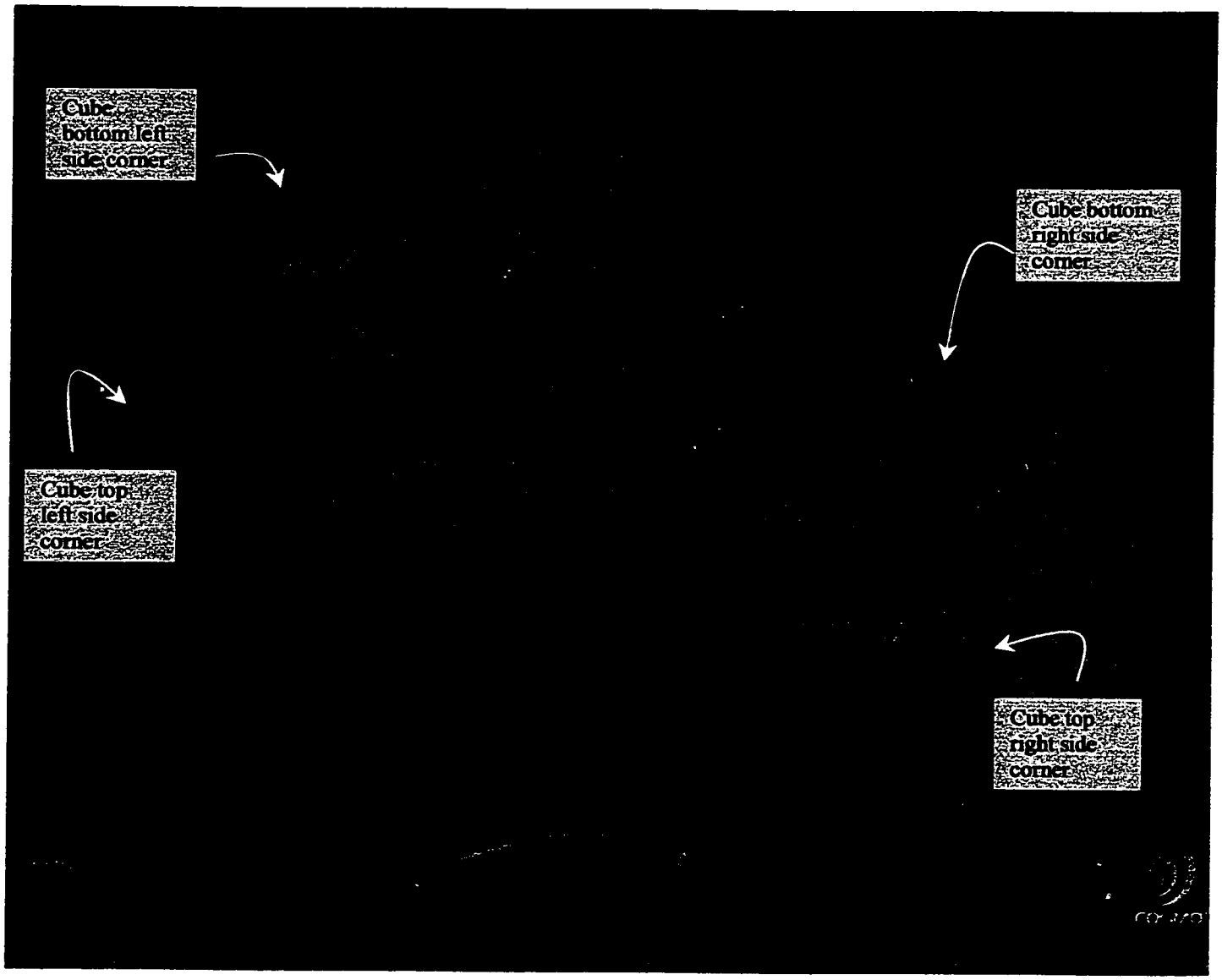
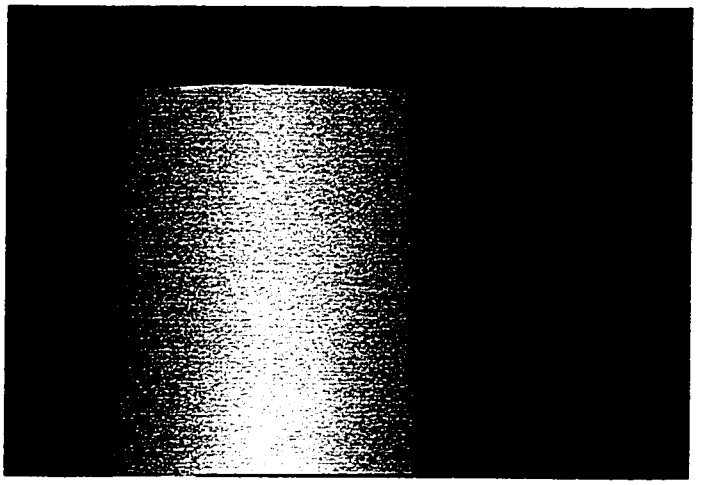
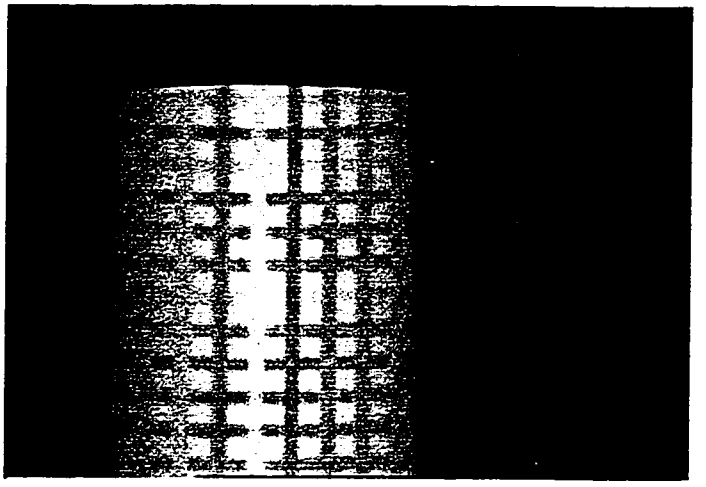


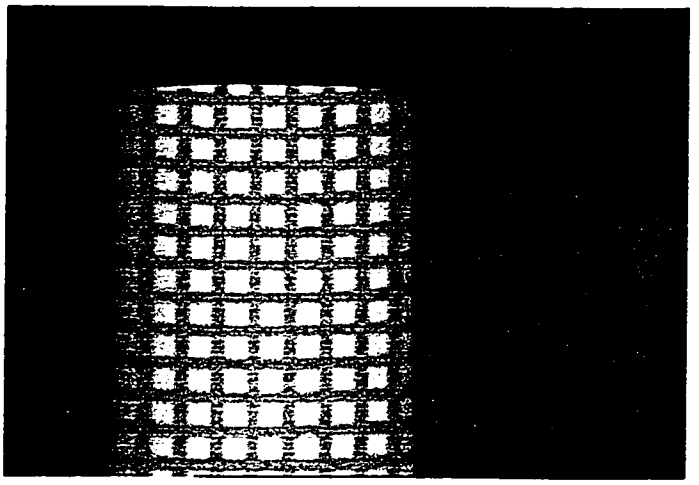
Figure 5.7 3D extracted cube



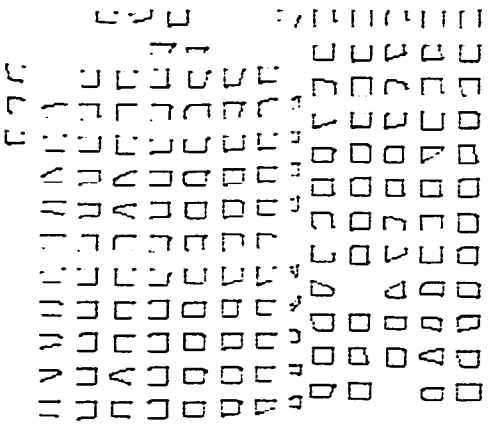
a) Introduction of cylinder onto the scene



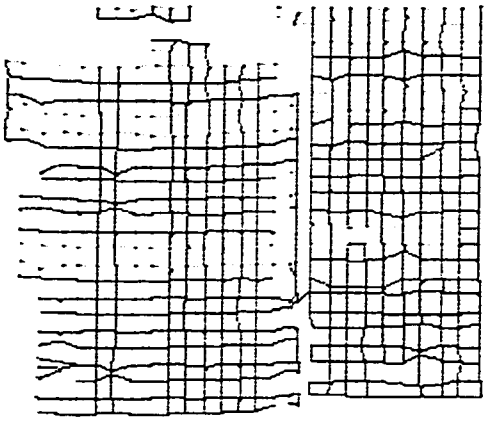
b) Projection of PRMVS grid onto the scene



c) Projection of uni-colored grid onto the scene



d) Extraction of grid edges



e) Reconstruction of the mesh

Figure 5.8 Grid extraction/reconstruction (with cylinder present in the scene)

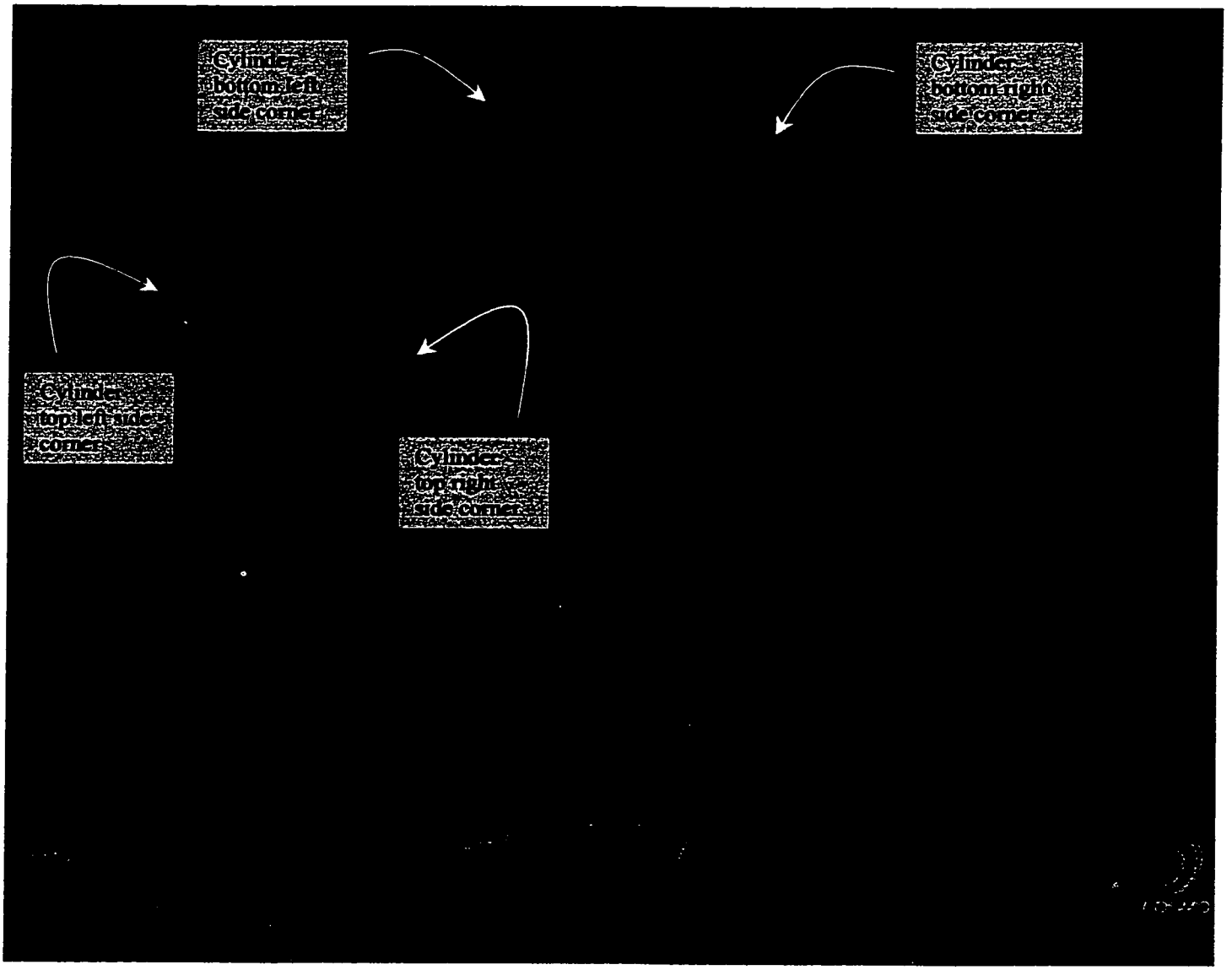


Figure 5.9 3D extracted cylinder

Chapter 6

Conclusion

The new pseudo-random color encoding method combined with structured light for object recognition from images proved to be successful. The experimental results demonstrated that it could work in any environment/scene regardless of its color where the object is placed, whereas other previous methods proved to work only in predefined scenes where the background color is known.

6.1 Achievements

- All components required for 3D-object recognition from 2D-images were implemented and tested for any generated errors.
- The real-time grid color selection is an original technique for selecting the best grid colors, which can best be extracted from the intended scene. By doing this, 3D-object extraction/recognition has no more need to be limited to a predefined colored scene. Objects can now be extracted in any scene.
- Multi-valued pseudo-random color encoded structured light proved to be a powerful and efficient way for resolving the "point identification" problem, which is normally encountered in structured light, in a colored scene.
- Recovering the edges of the extracted grid lines instead of their skeletons yielded the four corners instead of the center point of intersecting lines. This approach resulted in a 4:1 increase in the density of the points used for triangulation.

6.2 *Disadvantages and Limitations*

- Only the portions of the 3D object where the structured light can reach and be visible to the camera can be extracted and identified. Any parts of the object that has not been exposed to structured light or exposed to structured light but not visible to the camera will be lost.
- When the structured light is projected onto the object, the projected lines will be deformed (bent, broken, etc.) according to the object shape and orientation (explained in Chapter 4 and demonstrated in Chapter 5). Using colored pseudo-random encoded grid, problems may occur when identifying the correct neighbors of certain points which got shifted in position due to the presence of a 3D object. A search has to be made to track all the required points and their appropriate neighbors. Identifying each line separately (size, color, etc.) will reduce this search problem but has the disadvantage of not working in any colored environment.
- Finding the edges of the recovered grid requires a decision engine for reconnecting them and constructing a new grid from those edges. Whereas this step can be avoided when finding the skeleton of the grid.

6.3 *Future Work*

- By increasing the density of both column and row lines of the projected colored grid, a higher number of points can be obtained from the grid, thereby resulting in a higher resolution for the extracted object.
- A calibrated camera can be used for a more accurate absolute depth calculations.
- This work can be enhanced to function in real-time, thereby assisting mobile robots [33] in detecting objects in their path and identify their shapes and coordinates.

References:

[1] P. M. Will, K. S. Pennington, "Grid Coding: A Preprocessing Technique for Robot and Machine Vision", *In Proc. 2nd Int. Joint Conf. Artificial Intell.*, pages 66-68, Sep. 1971.

[2] U. R. Dhond, J. K. Aggarwal, "Structure from Stereo – A Review", *IEEE Trans. On Systems, Man and Cyber.*, vol. 19, no. 6, pp. 1489-1510, Nov./Dec. 1989.

[3] Y. F. Wang, A. Mitiche, J. K. Aggarwal, "Computation of Surface Orientation and Structure of Objects Using Grid Coding", *IEEE Trans. Pattern Anal. Machine Intell.*, vol, PAMI-9, no. 1, pp. 129-137, Jan. 1987.

[4] G. Hu, G. Stockman, "3-D Surface Solution Using Structured Light and Constraint Propagation", *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 11, no. 4, pp. 390-402, Apr. 1989.

[5] A. Rosenfeld, R. A. Hummel, S. W. Zucker, "Scene Labeling by Relaxation Operations", *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-6, pp. 420-443, 1976.

[6] Y. Sharai, "Recognition of Polyhedrals with a Range Finder", *Artificial Intell.*, vol. 4, no. 3, pp. 243-250, Oct. 1972.

[7] R. A. Jarvis, "A Perspective on Range Finding Techniques for Computer Vision", *IEEE Trans. Pattern Anal. Machine Intell.*, vol, PAMI-5, no. 2, pp. 122-139, Mar. 1983.

[8] K. L. Boyer A. C. Kak, "Color-Encoded Structured Light for Rapid Active Ranging", *IEEE Trans. Pattern anal. Machine Intell.*, vol. PAMI-9, no. 1, pp. 14-28, Jan. 1987.

[9] J. J. Le Moigne, A. M. Waxman, "Structured Light Patterns for Robot Mobility", *IEEE J. Robotics Automat*, vol. 4, no. 5, pp. 541-548, Oct. 1988.

[10] R. J. McEliece, "Finite Fields for Computer Scientists and Engineers", *Kluwer Academic Publishers*, 1987.

[11] L. G. Roberts, "Machine Perception of Three-Dimensional Solids", *In Optical and Electro-optical Information Processing*. Cambridge, MA: MIT Press, 1965.

[12] D. G. Lowe, T. O. Binford, "The Recovery of Three-Dimensional Structure from Image Curves", *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-7, no. 3, pp. 320-326, May. 1985.

[13] A. Wallace and S. Price. A Stereo Vision System. http://www.dai.ed.ac.uk/CVonline/LOCAL_COPIES/MARBLE/medium/stereo/system.htm

[14] R. Owens. Feature Matching. http://www.dai.ed.ac.uk/CVonline/LOCAL_COPIES/OWENS/LECT11/node5.html

[15] J. L. Posdemer, M. D. Altschuler, "Surface Measurement by Space-Encoded Projected Beam Systems", *Comput. Graphics Image Processing.* pp. 1 –17, 1982.

[16] H. G. Barrow, J. M. Tenenbaum, "Interpreting Line Drawings as Three-Dimensional Surfaces", *Artif. Intell.*, vol. 17, pp. 75-116, 1981.

[17] F. J. Mac Williams, N. J. A. Sloane, "Pseudo-Random Sequences and Arrays", *Proc. IEEE.* vol. 64, no. 12. Dec. 1976.

[18] E. M. Petriu, J. S. Basran, F. C. A. Groen, "Automated Guided Vehicle Position Recovery", *IEEE Trans. Instr. Measur.*, vol 39, no. 1, Feb. 1990.

[19] H. Khalfallah, E. M. Petriu, F. C. A. Groen, "Visual Position Recovery for an Automated Guided Vehicle", *IEEE Trans. Instr. Measur.*, vol. 41, no. 6, Dec. 1992.

[20] E. M. Petriu, W. S. McMath, S. K. Yeung, N. Trif, T. Bieseman, "Two-Dimensional Position Recovery for a Free-Ranging Automated Guided Vehicle", *IEEE Trans. Instr. Measur.*, vol. 42, no. 3, Jun. 1993.

[21] P. Vuylsteke, A. Oosternlink, "Range Image Acquisition with a Single Binary Encoded Light Pattern", *IEEE Trans. Pattern anal. Machine Intell.*, vol. 12, no. 2, pp. 148-164, Feb. 1990.

[22] D. Rosenberg, M. D. Levine, S. W. Zucker, "Computing Relative Depth Relationships from Occlusion Cues", *In Proc. 4th Int. Joint Conf. Pattern Recognition, Kyoto, Japan*, Nov. 7-10, 1978, pp. 765-769.

[23] S. E. Umbaugh, "Computer Vision and Image Processing, a Practical Approach using CVIPtools", *Prentice-Hall PTR*, Upper Saddle River, N. J. 1998.

[24] A. K. Jain, "Fundamentals of Digital Image Processing", *Prentice-Hall, Inc.*, Englewood, N. J. 1989.

[25] D. C. Buchthal, D. E. Cameron, "Modern Abstract Algebra", *PWS Publishers.* 1987.

[26] S. Lin, D. J. Costello, "Error Control Coding, Fundamentals and Applications", *Prentice-Hall, Inc.*, Englewood Cliffs, N. J. 1983.

[27] D. H. Ballard, C. M. Brown, "Computer Vision", *Prentice-Hall, Inc.*, Englewood Cliffs, N. J., 1982.

[28] S.W. Golumb, "Digital Communication with Space Applications", *Prentice-Hall, Inc.*, Englewood Cliffs, N. J. 1966.

[29] C. Balza, A. Fromageot, M. Maniere, "Four-Level Pseudo-Random Sequences", *Electron. Lett.*, vol.3, pp. 313-315, 1967.

[30] E.M. Petriu, "Absolute Type Position Transducers using Pseudo-Random Encoding", *IEEE Trans. Instrum. Meas.*, vol. IM-36, no. 4, pp.950-955, Feb. 1987.

[31] N. Trif, "Model-Based Visual Recognition of 3D Objects using Pseudo-Random Grid Coding", Masters thesis, University of Ottawa, May 1993.

[32] D.H. Green, I.S. Taylor, "Irreducible Polynomials over Composite Galois Fields and their Application in Coding Techniques", *Proc. Inst. Elec. Eng.*, vol. 121, pp. 935-939, 1974.

[33] **Z. Sakr**, E. Petriu, P. Wide, "Fuzzy Controller for a Hexaped Robot", *Proc. VIMS'99, IEEE Workshop on Virtual and Intelligent Measurement Systems*, pp. 42-46, Venice, Italy, May 1999.

[34] P. Vuylsteke, A. Oosterlink, "Range Image Acquisition with a Single Binary-Encoded Light Pattern", *IEEE Trans. Pattern Anal. Machine Intell*, vol. 12, no. 2, pp. 148-164, Feb. 1990.

[35] H. R. Everett, "Sensors for Mobile Robots – Theory and Application", *A K Peters, Ltd.*, Wellesley, MA., 1995.

[36] E. Petriu, **Z. Sakr**, H.J.W. Spoelder, A. Moica, "Object Recognition Using Pseudo-Random Color Encoded Structured Light", *Proc. IMTC/2000, IEEE Instrum. Meas. Technol. Conf.*, Baltimore, MD, May 2000.

[37] E. Petriu, S.R. Das, N. Trif, S.K. Yeung, "Pseudo-Random Encoding for Structured Light Applications", *Proc. ISCA 13th Intl. Conf. Computers and Their Applications*, pp. 287-290, Honolulu, Hawaii, 1998.

[38] P. H. S. Torr, D. W. Murray, "The Development and Comparison of Robust Methods for Estimating the Fundamental Matrix", *Int. Journal of Computer Vision*, vol. 24, no. 3, pp. 271-300, 1997.

[39] Epipolar Geometry, http://jebara.www.media.mit.edu/people/jebara/sfm/node8.html

[40] P. Lavoie, D. Ionescu, E. Petriu, "3-D Object Model Recovery From 2-D Images Using Structured Light", *Proc. IMTC/96, IEEE Instrum. Meas. Technol. Conf.*, pp.377-382, Brussels, Belgium, 1996.

[41] E. Petriu, N. Trif, W.S. McMath, S.K. Yeung, "Automated Guided Vehicle Position Recovery Using Pseudo-Random Grid Encoding", *Proc. Int. Conf. Intell. Autonomous Syst. IAS-3*, pp.359-367, Pittsburgh, PA, 1993.

[42] Introduction to Hue, Saturation and Brightness, http://www.thetech.org/hyper/color/hsb/circles.html

[43] J. R. Ball, A. H. Spittle, H. T. Liu, "High-Speed m-Sequence Generation: a Further Note," *Electron. Lett.*, vol. 11, pp. 107-108, 1975.

[44] D. E. Carter, "On The Generation of Pseudo-Noise Codes," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 10, pp. 898-899, 1974.

[45] I. G. Cumming, "Autocorrelation Function and Spectrum of a Filtered, Pseudo-Random Binary Sequence," *Proc. Inst. Elect. Eng.*, vol. 114, pp. 1360-1362, 1967.

[46] S. Fredricsson, "Pseudo-Randomness Properties of Binary Shift Register Sequences," *IEEE Trans. Inform. Theory*, vol. 21, pp. 115-120, 1975.

[47] J. T. Harvey, "High-Speed m-Sequence Generation," *Electron. Lett.*, vol. 10, pp. 480-481, 1974.

[48] P. D. Roberts, R. H. Davis, "Statistical Properties of Smoothed Maximal-Length Linear Binary Sequences," *Proc. Inst. Elec. Eng.*, vol. 113, pp. 190-196, 1966.

[49] T. C. Bartee, D. I. Schneider, "Computation with Finite Fields," *Infom. Contr.*, vol. 6, pp. 79-98, 1963.

[50] J. T. B. Beard, "Computing in GF(q)," *Math. Comput.*, vol. 28, pp. 1159-1166, 1974.

[51] E. R. Berlekamp, "Algebraic Coding Theory". New York: McGrawHill, 1968.

[52] C. M. Holt, A. Stewart, M. Clint, R. H. Perrott, "An Improved Parallel Thinning Algorithm," *Comm. ACM*, vol. 30, no. 2, pp. 156-160, 1987.

[53] T. Y. Zhang, C. Y. Suen, "A Fast Parallel Algorithm for Thinning Digital Patterns," *Comm. ACM*, vol. 27, no.3, pp. 236-239, 1984.

[54] R. W. Hall, "Fast Parallel Thinning Algorithms: Parallel Speed and Connectivity Preservation," *Comm. ACM*, vol. 32, no. 1, pp. 124-131, 1989.

[55] Edge Detection, http://www.ee.bgu.ac.il/~greg/graphics/special.html

[56] Color - RGB, http://home.flash.net/~drj2142/rgb.html

Appendix A

Primitive Polynomial over GF(q)[17]

m	$GF(2)$	$GF(3)$	$GF(2^2)$	$GF(2^3)$
1	$x+1$	$x+1$	$x+w$	$x+\alpha$
2	x^2+x+1	x^2+x+2	x^2+x+w	$x^2+\alpha x+\alpha$
3	x^3+x+1	x^3+2x+1	x^3+x^2+x+w	$x^3+x+\alpha$
4	x^4+x+1	x^4+x+2	$x^4+x^2+wx+w^2$	$x^4+x+\alpha^3$
5	x^5+x^2+1	x^5+2x+1	x^5+x+w	$x^5+x^2+x+\alpha^3$
6	x^6+x+1	x^6+x+2	x^6+x^2+x+w	$x^6+x+\alpha$
7	x^7+x+1	$x^7+x^6+x^4+1$	$x^7+x^2+wx+w^2$	$x^7+x^2+\alpha x+\alpha^3$
8	$x^8+x^6+x^5+x+1$	x^8+x^5+2	x^8+x^3+x+w	
9	x^9+x^4+1	$x^9+x^7+x^5+1$	x^9+x^2+x+w	
10	$x^{10}+x^3+1$	$x^{10}+x^9+x^7+2$	$x^{10}+x^3+w(x^2+x+1)$	
11	$x^{11}+x^2+1$			
12	$x^{12}+x^7+x^4+x^3+x+1$			
13	$x^{13}+x^4+x^3+x+1$			
14	$x^{14}+x^{12}+x^{11}+x+1$			
15	$x^{15}+x+1$			

where

w - is a generating element for GF(2^2)

α - is a generating element for GF(2^3).

Appendix B

The elements of some Galois fields [33], [10]

0	$-\infty$
1	0

Table B.1 The elements of GF(2)

00	$-\infty$
10	0
01	1
11	2

Table B.2 The elements of GF(2^2) generated by h(x) = x^2+x+1

000	$-\infty$
100	0
010	1
001	2
110	3
011	4
111	5
101	6

Table B.3 The elements of GF(2^3) generated by h(x) = x^3+x+1

0000	$-\infty$
1000	0
0100	1
0010	2
0001	3
1100	4
0110	5
0011	6
1101	7
1010	8
0101	9
1110	10
0111	11
1111	12
1011	13
1001	14

Table B.4 The elements of the GF(2^4) generated by $h(x) = x^4+x+1$

00000	$-\infty$	11111	15
10000	0	11011	16
01000	1	11001	17
00100	2	11000	18
00010	3	01100	19
00001	4	00110	20
10100	5	00011	21
01010	6	10101	22
00101	7	11110	23
10110	8	01111	24
01011	9	10011	25
10001	10	11101	26
11100	11	11010	27
01110	12	01101	28
00111	13	10010	29
10111	14	01001	30

Table B.5 The elements of GF(2^5) generated by $h(x) = x^5+x^2+1$

000000	-∞	110111	21	111011	43
100000	0	101011	22	101101	44
010000	1	100101	23	100110	45
001000	2	100010	24	010011	46
000100	3	010001	25	111001	47
000010	4	111000	26	101100	48
000001	5	011100	27	010110	49
110000	6	001110	28	001011	50
011000	7	000111	29	110101	51
001100	8	110011	30	101010	52
000110	9	101001	31	010101	53
000011	10	100100	32	111010	54
110001	11	010010	33	011101	55
101000	12	001001	34	111110	56
010100	13	110100	35	011111	57
001010	14	011010	36	111111	58
000101	15	001101	37	101111	59
110010	16	110110	38	100111	60
011001	17	011011	39	100011	61
111100	18	111101	40	100001	62
011110	19	101110	41		
001111	20	010111	42		

Table B.6 The elements of GF(2^6) generated by $h(x) = x^6+x+1$

Appendix C

3D coordinates of extracted cube (chapter 5):

Blue coordinate:(51, 22, 9.32), Position: (0, 0)
Blue coordinate:(104, 23, 10.88), Position: (0, 3)
Blue coordinate:(121, 25, 11.04), Position: (0, 4)
Blue coordinate:(138, 25, 11.41), Position: (0, 5)
Blue coordinate:(156, 24, 10.49), Position: (0, 6)
Blue coordinate:(230, 10, 0.66), Position: (0, 11)
Blue coordinate:(51, 38,9.32), Position: (1, 0)
Blue coordinate:(69, 43, 9.43), Position: (1, 1)
Blue coordinate:(86, 44, 10.91), Position: (1, 2)
Blue coordinate:(104, 40, 10.88), Position: (1, 3)
Blue coordinate:(121, 41,11.04), Position: (1, 4)
Blue coordinate:(138, 43, 12.51), Position: (1, 5)
Blue coordinate:(154, 42, 10.49), Position: (1, 6)
Blue coordinate:(170, 39, 9.57), Position: (1, 7)
Blue coordinate:(214, 27, 2.73), Position: (1, 10)
Blue coordinate:(231, 27, 0.94), Position: (1, 11)
Blue coordinate:(50, 55, 9.32), Position: (2, 0)
Blue coordinate:(69, 55, 9.43), Position: (2, 1)
Blue coordinate:(86, 55, 9.57), Position: (2, 2)
Blue coordinate:(103, 57,10.08), Position: (2, 3)
Blue coordinate:(120, 57,10.11), Position: (2, 4)
Blue coordinate:(138, 58, 11.96), Position: (2, 5)
Blue coordinate:(154, 57, 10.49), Position: (2, 6)
Blue coordinate:(186, 52, 6.57), Position: (2, 8)

Blue coordinate:(200, 50, 5.35), Position: (2, 9)
Blue coordinate:(214, 43, 1.50), Position: (2, 10)
Blue coordinate:(232, 44, 1.48), Position: (2, 11)
Blue coordinate:(50, 71, 8.10), Position: (3, 0)
Blue coordinate:(68, 72, 9.02), Position: (3, 1)
Blue coordinate:(86, 72, 9.02), Position: (3, 2)
Blue coordinate:(103, 72, 9.43), Position: (3, 3)
Blue coordinate:(120, 73, 9.95), Position: (3, 4)
Blue coordinate:(137, 75, 11.04), Position: (3, 5)
Blue coordinate:(184, 69, 6.63), Position: (3, 8)
Blue coordinate:(202, 66, 5.33), Position: (3, 9)
Blue coordinate:(214, 61, 1.03), Position: (3, 10)
Blue coordinate:(230, 61, 0.00), Position: (3, 11)
Blue coordinate:(248, 60, 0.00), Position: (3, 12)
Blue coordinate:(50, 88, 7.65), Position: (4, 0)
Blue coordinate:(68, 88, 7.99), Position: (4, 1)
Blue coordinate:(86, 88, 9.02), Position: (4, 2)
Blue coordinate:(103, 89, 9.43), Position: (4, 3)
Blue coordinate:(120, 89, 10.30), Position: (4, 4)
Blue coordinate:(137, 91, 10.49), Position: (4, 5)
Blue coordinate:(156, 89, 10.41), Position: (4, 6)
Blue coordinate:(184, 85, 5.71), Position: (4, 8)
Blue coordinate:(200, 83, 4.78), Position: (4, 9)
Blue coordinate:(232, 78, 1.33), Position: (4, 11)
Blue coordinate:(68, 104, 8.49), Position: (5, 1)
Blue coordinate:(85, 105, 9.03), Position: (5, 2)
Blue coordinate:(102, 105, 9.35), Position: (5, 3)
Blue coordinate:(120, 106, 9.41), Position: (5, 4)
Blue coordinate:(136, 108, 9.57), Position: (5, 5)
Blue coordinate:(184, 101, 5.63), Position: (5, 8)
Blue coordinate:(200, 99, 3.75), Position: (5, 9)

Blue coordinate:(213, 100, 3.38), Position: (5, 10)
Blue coordinate:(230, 94, 0.66), Position: (5, 11)
Blue coordinate:(248, 95, 0.10), Position: (5, 12)
Blue coordinate:(49, 124, 8.30), Position: (6, 0)
Blue coordinate:(68, 120, 8.37), Position: (6, 1)
Blue coordinate:(84, 122, 8.82), Position: (6, 2)
Blue coordinate:(102, 122, 9.02), Position: (6, 3)
Blue coordinate:(118, 126, 10.72), Position: (6, 4)
Blue coordinate:(136, 123, 9.57), Position: (6, 5)
Blue coordinate:(170, 120, 7.99), Position: (6, 7)
Blue coordinate:(198, 118, 4.83), Position: (6, 9)
Blue coordinate:(214, 111, 2.39), Position: (6, 10)
Blue coordinate:(230, 111, 0.66), Position: (6, 11)
Blue coordinate:(247, 112, 0.66), Position: (6, 12)
Blue coordinate:(49, 137, 7.17), Position: (7, 0)
Blue coordinate:(67, 137, 7.56), Position: (7, 1)
Blue coordinate:(84, 138, 8.04), Position: (7, 2)
Blue coordinate:(102, 138, 8.49), Position: (7, 3)
Blue coordinate:(119, 139, 8.49), Position: (7, 4)
Blue coordinate:(214, 129, 1.33), Position: (7, 10)
Blue coordinate:(246, 129, 1.48), Position: (7, 12)
Blue coordinate:(48, 153, 5.63), Position: (8, 0)
Blue coordinate:(67, 153, 7.19), Position: (8, 1)
Blue coordinate:(84, 154, 7.38), Position: (8, 2)
Blue coordinate:(101, 155, 8.06), Position: (8, 3)
Blue coordinate:(118, 158, 8.52), Position: (8, 4)
Blue coordinate:(136, 156, 9.02), Position: (8, 5)
Blue coordinate:(247, 146, 0.66), Position: (8, 12)
Blue coordinate:(84, 170, 6.63), Position: (9, 2)
Blue coordinate:(101, 171, 8.10), Position: (9, 3)
Blue coordinate:(118, 172, 8.49), Position: (9, 4)

Blue coordinate:(152, 175, 9.57), Position: (9, 6)
Blue coordinate:(184, 118, 5.96), Position: (9, 8)
Blue coordinate:(212, 162, 0.63), Position: (9, 10)
Blue coordinate:(230, 162, 0.66), Position: (9, 11)
Blue coordinate:(247, 163, 0.42), Position: (9, 12)
Blue coordinate:(196, 179, 0.00), Position: (10, 9)
Blue coordinate:(212, 180, 0.00), Position: (10, 10)
Blue coordinate:(230, 179, 0.66), Position: (10, 11)
Blue coordinate:(248, 180, 0.00), Position: (10, 12)
Blue coordinate:(42, 196, 0.00), Position: (11, 0)
Blue coordinate:(195, 196, 0.94), Position: (11, 9)
Blue coordinate:(212, 197, 0.00), Position: (11, 10)
Blue coordinate:(230, 196, 0.66), Position: (11, 11)
Blue coordinate:(247, 199, 1.48), Position: (11, 12)

Dark blue coordinate:(61, 22, 9.32), Position: (0, 0)
Dark blue coordinate:(112, 24, 10.49), Position: (0, 3)
Dark blue coordinate:(130, 26, 12.88), Position: (0, 4)
Dark blue coordinate:(147, 26, 11.96), Position: (0, 5)
Dark blue coordinate:(161, 23, 10.49), Position: (0, 6)
Dark blue coordinate:(239, 10, 0.66), Position: (0, 11)
Dark blue coordinate:(60, 38, 9.32), Position: (1, 0)
Dark blue coordinate:(77, 39, 9.43), Position: (1, 1)
Dark blue coordinate:(95, 39, 9.44), Position: (1, 2)
Dark blue coordinate:(112, 40, 10.43), Position: (1, 3)
Dark blue coordinate:(129, 41, 10.75), Position: (1, 4)
Dark blue coordinate:(146, 42, 11.04), Position: (1, 5)
Dark blue coordinate:(161, 40, 10.12), Position: (1, 6)
Dark blue coordinate:(177, 37, 7.17), Position: (1, 7)
Dark blue coordinate:(222, 29, 1.48), Position: (1, 10)
Dark blue coordinate:(239, 27, 0.00), Position: (1, 11)

Dark blue coordinate:(60, 54, 9.32), Position: (2, 0)

Dark blue coordinate:(77, 55, 9.43), Position: (2, 1)

Dark blue coordinate:(95, 56, 10.88), Position: (2, 2)

Dark blue coordinate:(112, 58, 10.11), Position: (2, 3)

Dark blue coordinate:(129, 57, 10.67), Position: (2, 4)

Dark blue coordinate:(145, 58, 11.04), Position: (2, 5)

Dark blue coordinate:(161, 57, 10.12), Position: (2, 6)

Dark blue coordinate:(192, 52, 6.83), Position: (2, 8)

Dark blue coordinate:(207, 49, 3.87), Position: (2, 9)

Dark blue coordinate:(221, 44, 1.49), Position: (2, 10)

Dark blue coordinate:(238, 44, 1.48), Position: (2, 11)

Dark blue coordinate:(59, 71, 9.02), Position: (3, 0)

Dark blue coordinate:(77, 71, 8.49), Position: (3, 1)

Dark blue coordinate:(95, 73, 10.36), Position: (3, 2)

Dark blue coordinate:(111, 73, 10.36), Position: (3, 3)

Dark blue coordinate:(128, 74, 10.49), Position: (3, 4)

Dark blue coordinate:(146, 75, 9.43), Position: (3, 5)

Dark blue coordinate:(191, 68, 5.35), Position: (3, 8)

Dark blue coordinate:(207, 65, 4.21), Position: (3, 9)

Dark blue coordinate:(221, 61, 1.99), Position: (3, 10)

Dark blue coordinate:(239, 61, 1.99), Position: (3, 11)

Dark blue coordinate:(256, 61, 0.66), Position: (3, 12)

Dark blue coordinate:(59, 87, 8.49), Position: (4, 0)

Dark blue coordinate:(77, 88, 8.49), Position: (4, 1)

Dark blue coordinate:(95, 89, 10.36), Position: (4, 2)

Dark blue coordinate:(111, 89, 9.43), Position: (4, 3)

Dark blue coordinate:(128, 90, 9.72), Position: (4, 4)

Dark blue coordinate:(146, 91, 9.86), Position: (4, 5)

Dark blue coordinate:(161, 89, 9.87), Position: (4, 6)

Dark blue coordinate:(191, 85, 5.35), Position: (4, 8)

Dark blue coordinate:(207, 83, 3.32), Position: (4, 9)

Dark blue coordinate:(239, 78, 0.66), Position: (4, 11)
Dark blue coordinate:(77, 106, 8.92), Position: (5, 1)
Dark blue coordinate:(94, 106, 9.03), Position: (5, 2)
Dark blue coordinate:(111, 105, 9.23), Position: (5, 3)
Dark blue coordinate:(129, 111, 10.01), Position: (5, 4)
Dark blue coordinate:(145, 107, 9.05), Position: (5, 5)
Dark blue coordinate:(191, 102, 5.35), Position: (5, 8)
Dark blue coordinate:(205, 99, 2.73), Position: (5, 9)
Dark blue coordinate:(221, 95, 0.00), Position: (5, 10)
Dark blue coordinate:(239, 95, 0.66), Position: (5, 11)
Dark blue coordinate:(255, 95, 0.00), Position: (5, 12)
Dark blue coordinate:(59, 120, 7.50), Position: (6, 0)
Dark blue coordinate:(76, 121, 8.38), Position: (6, 1)
Dark blue coordinate:(93, 121, 8.83), Position: (6, 2)
Dark blue coordinate:(110, 122, 9.02), Position: (6, 3)
Dark blue coordinate:(128, 123, 10.36), Position: (6, 4)
Dark blue coordinate:(144, 123, 9.02), Position: (6, 5)
Dark blue coordinate:(175, 120, 5.18), Position: (6, 7)
Dark blue coordinate:(205, 115, 1.48), Position: (6, 9)
Dark blue coordinate:(221, 111, 0.00), Position: (6, 10)
Dark blue coordinate:(239, 112, 0.66), Position: (6, 11)
Dark blue coordinate:(255, 112, 0.00), Position: (6, 12)
Dark blue coordinate:(59, 137, 7.99), Position: (7, 0)
Dark blue coordinate:(75, 137, 7.99), Position: (7, 1)
Dark blue coordinate:(93, 137, 8.07), Position: (7, 2)
Dark blue coordinate:(110, 138, 8.92), Position: (7, 3)
Dark blue coordinate:(127, 139, 8.53), Position: (7, 4)
Dark blue coordinate:(221, 129, 0.00), Position: (7, 10)
Dark blue coordinate:(255, 128, 0.66), Position: (7, 12)
Dark blue coordinate:(58, 153, 6.57), Position: (8, 0)
Dark blue coordinate:(75, 153, 7.19), Position: (8, 1)

Dark blue coordinate:(93, 155, 7.39), Position: (8, 2)
Dark blue coordinate:(110, 155, 8.10), Position: (8, 3)
Dark blue coordinate:(127, 155, 8.56), Position: (8, 4)
Dark blue coordinate:(144, 156, 7.50), Position: (8, 5)
Dark blue coordinate:(255, 145, 0.66), Position: (8, 12)
Dark blue coordinate:(89, 170, 5.93), Position: (9, 2)
Dark blue coordinate:(110, 173, 9.43), Position: (9, 3)
Dark blue coordinate:(127, 172, 7.99), Position: (9, 4)
Dark blue coordinate:(159, 172, 6.12), Position: (9, 6)
Dark blue coordinate:(189, 117, 5.54), Position: (9, 8)
Dark blue coordinate:(220, 163, 0.63), Position: (9, 10)
Dark blue coordinate:(238, 163, 0.38), Position: (9, 11)
Dark blue coordinate:(255, 163, 0.42), Position: (9, 12)
Dark blue coordinate:(203, 179, 3.38), Position: (10, 9)
Dark blue coordinate:(221, 179, 0.66), Position: (10, 10)
Dark blue coordinate:(239, 180, 0.33), Position: (10, 11)
Dark blue coordinate:(255, 179, 0.48), Position: (10, 12)
Dark blue coordinate:(50, 200, 0.00), Position: (11, 0)
Dark blue coordinate:(204, 200, 0.00), Position: (11, 9)
Dark blue coordinate:(221, 198, 0.66), Position: (11, 10)
Dark blue coordinate:(239, 197, 0.66), Position: (11, 11)
Dark blue coordinate:(255, 196, 2.73), Position: (11, 12)

Dark yellow coordinate:(61, 29, 9.32), Position: (0, 0)
Dark yellow coordinate:(113, 30, 10.88), Position: (0, 3)
Dark yellow coordinate:(130, 32, 11.81), Position: (0, 4)
Dark yellow coordinate:(147, 33, 12.51), Position: (0, 5)
Dark yellow coordinate:(161, 30, 10.49), Position: (0, 6)
Dark yellow coordinate:(239, 14, 1.99), Position: (0, 11)
Dark yellow coordinate:(60, 45, 9.32), Position: (1, 0)
Dark yellow coordinate:(77, 47, 9.43), Position: (1, 1)

Dark yellow coordinate:(95, 47, 9.44), Position: (1, 2)
Dark yellow coordinate:(112, 48, 11.41), Position: (1, 3)
Dark yellow coordinate:(130, 43, 10.75), Position: (1, 4)
Dark yellow coordinate:(146, 49, 13.56), Position: (1, 5)
Dark yellow coordinate:(161, 47, 9.57), Position: (1, 6)
Dark yellow coordinate:(177, 44, 10.10), Position: (1, 7)
Dark yellow coordinate:(219, 34, 1.33), Position: (1, 10)
Dark yellow coordinate:(239, 33, 0.66), Position: (1, 11)
Dark yellow coordinate:(60, 62, 9.32), Position: (2, 0)
Dark yellow coordinate:(77, 63, 9.95), Position: (2, 1)
Dark yellow coordinate:(95, 63, 10.36), Position: (2, 2)
Dark yellow coordinate:(112, 64, 10.11), Position: (2, 3)
Dark yellow coordinate:(129, 65, 10.67), Position: (2, 4)
Dark yellow coordinate:(146, 65, 11.41), Position: (2, 5)
Dark yellow coordinate:(161, 63, 9.57), Position: (2, 6)
Dark yellow coordinate:(191, 58, 5.35), Position: (2, 8)
Dark yellow coordinate:(207, 56, 3.87), Position: (2, 9)
Dark yellow coordinate:(221, 51, 1.50), Position: (2, 10)
Dark yellow coordinate:(237, 51, 1.33), Position: (2, 11)
Dark yellow coordinate:(59, 79, 9.02), Position: (3, 0)
Dark yellow coordinate:(77, 79, 9.02), Position: (3, 1)
Dark yellow coordinate:(95, 79, 9.86), Position: (3, 2)
Dark yellow coordinate:(111, 81, 10.49), Position: (3, 3)
Dark yellow coordinate:(129, 81, 10.88), Position: (3, 4)
Dark yellow coordinate:(146, 81, 10.88), Position: (3, 5)
Dark yellow coordinate:(191, 75, 5.35), Position: (3, 8)
Dark yellow coordinate:(207, 72, 4.21), Position: (3, 9)
Dark yellow coordinate:(221, 69, 1.99), Position: (3, 10)
Dark yellow coordinate:(239, 68, 0.66), Position: (3, 11)
Dark yellow coordinate:(255, 68, 0.00), Position: (3, 12)
Dark yellow coordinate:(59, 95, 8.49), Position: (4, 0)

Dark yellow coordinate:(77, 95, 9.28), Position: (4, 1)
Dark yellow coordinate:(94, 96, 9.43), Position: (4, 2)
Dark yellow coordinate:(111, 97, 9.86), Position: (4, 3)
Dark yellow coordinate:(129, 97, 9.72), Position: (4, 4)
Dark yellow coordinate:(146, 97, 9.87), Position: (4, 5)
Dark yellow coordinate:(160, 96, 9.87), Position: (4, 6)
Dark yellow coordinate:(191, 91, 4.78), Position: (4, 8)
Dark yellow coordinate:(205, 89, 2.73), Position: (4, 9)
Dark yellow coordinate:(239, 85, 0.66), Position: (4, 11)
Dark yellow coordinate:(77, 111, 8.44), Position: (5, 1)
Dark yellow coordinate:(94, 112, 9.03), Position: (5, 2)
Dark yellow coordinate:(111, 113, 9.23), Position: (5, 3)
Dark yellow coordinate:(129, 112, 10.01), Position: (5, 4)
Dark yellow coordinate:(145, 115, 10.49), Position: (5, 5)
Dark yellow coordinate:(189, 108, 4.19), Position: (5, 8)
Dark yellow coordinate:(205, 106, 2.73), Position: (5, 9)
Dark yellow coordinate:(221, 102, 0.66), Position: (5, 10)
Dark yellow coordinate:(239, 102, 0.66), Position: (5, 11)
Dark yellow coordinate:(255, 102, 0.66), Position: (5, 12)
Dark yellow coordinate:(59, 127, 7.99), Position: (6, 0)
Dark yellow coordinate:(76, 128, 8.38), Position: (6, 1)
Dark yellow coordinate:(93, 129, 8.83), Position: (6, 2)
Dark yellow coordinate:(111, 129, 10.49), Position: (6, 3)
Dark yellow coordinate:(128, 130, 9.43), Position: (6, 4)
Dark yellow coordinate:(145, 131, 9.95), Position: (6, 5)
Dark yellow coordinate:(175, 126, 5.71), Position: (6, 7)
Dark yellow coordinate:(205, 122, 2.39), Position: (6, 9)
Dark yellow coordinate:(221, 119, 0.00), Position: (6, 10)
Dark yellow coordinate:(239, 118, 0.94), Position: (6, 11)
Dark yellow coordinate:(255, 119, 0.66), Position: (6, 12)
Dark yellow coordinate:(58, 144, 7.05), Position: (7, 0)

Dark yellow coordinate:(75, 145, 7.99), Position: (7, 1)
Dark yellow coordinate:(93, 145, 8.07), Position: (7, 2)
Dark yellow coordinate:(110, 146, 8.49), Position: (7, 3)
Dark yellow coordinate:(127, 147, 8.53), Position: (7, 4)
Dark yellow coordinate:(220, 135, 1.48), Position: (7, 10)
Dark yellow coordinate:(255, 136, 0.66), Position: (7, 12)
Dark yellow coordinate:(58, 154, 4.69), Position: (8, 0)
Dark yellow coordinate:(75, 161, 7.21), Position: (8, 1)
Dark yellow coordinate:(93, 161, 7.38), Position: (8, 2)
Dark yellow coordinate:(110, 162, 8.10), Position: (8, 3)
Dark yellow coordinate:(127, 163, 8.56), Position: (8, 4)
Dark yellow coordinate:(144, 163, 10.12), Position: (8, 5)
Dark yellow coordinate:(255, 153, 0.66), Position: (8, 12)
Dark yellow coordinate:(93, 175, 7.17), Position: (9, 2)
Dark yellow coordinate:(110, 175, 6.83), Position: (9, 3)
Dark yellow coordinate:(127, 180, 9.02), Position: (9, 4)
Dark yellow coordinate:(159, 178, 7.17), Position: (9, 6)
Dark yellow coordinate:(189, 125, 5.53), Position: (9, 8)
Dark yellow coordinate:(221, 170, 0.63), Position: (9, 10)
Dark yellow coordinate:(237, 170, 0.33), Position: (9, 11)
Dark yellow coordinate:(255, 170, 0.42), Position: (9, 12)
Dark yellow coordinate:(204, 187, 1.33), Position: (10, 9)
Dark yellow coordinate:(221, 187, 0.66), Position: (10, 10)
Dark yellow coordinate:(239, 186, 0.66), Position: (10, 11)
Dark yellow coordinate:(255, 187, 2.73), Position: (10, 12)
Dark yellow coordinate:(50, 201, 0.00), Position: (11, 0)
Dark yellow coordinate:(204, 201, 1.99), Position: (11, 9)
Dark yellow coordinate:(221, 205, 0.00), Position: (11, 10)
Dark yellow coordinate:(239, 204, 1.99), Position: (11, 11)
Dark yellow coordinate:(255, 205, 0.66), Position: (11, 12)

Magenta coordinate:(51, 28, 0.32), Position: (0, 0)
Magenta coordinate:(104, 31, 11.41), Position: (0, 3)
Magenta coordinate:(122, 33, 11.96), Position: (0, 4)
Magenta coordinate:(138, 33, 13.08), Position: (0, 5)
Magenta coordinate:(154, 30, 10.49), Position: (0, 6)
Magenta coordinate:(230, 17, 0.66), Position: (0, 11)
Magenta coordinate:(51, 45, 9.32), Position: (1, 0)
Magenta coordinate:(69, 46, 9.43), Position: (1, 1)
Magenta coordinate:(86, 47, 10.91), Position: (1, 2)
Magenta coordinate:(104, 48, 10.88), Position: (1, 3)
Magenta coordinate:(120, 48, 11.04), Position: (1, 4)
Magenta coordinate:(138, 49, 11.96), Position: (1, 5)
Magenta coordinate:(154, 48, 10.49), Position: (1, 6)
Magenta coordinate:(170, 45, 8.49), Position: (1, 7)
Magenta coordinate:(214, 35, 0.94), Position: (1, 10)
Magenta coordinate:(230, 34, 0.00), Position: (1, 11)
Magenta coordinate:(50, 61, 9.32), Position: (2, 0)
Magenta coordinate:(68, 62, 9.34), Position: (2, 1)
Magenta coordinate:(86, 63, 9.57), Position: (2, 2)
Magenta coordinate:(104, 64, 10.08), Position: (2, 3)
Magenta coordinate:(120, 65, 10.11), Position: (2, 4)
Magenta coordinate:(138, 66, 11.96), Position: (2, 5)
Magenta coordinate:(154, 64, 9.57), Position: (2, 6)
Magenta coordinate:(184, 59, 7.73), Position: (2, 8)
Magenta coordinate:(200, 57, 4.78), Position: (2, 9)
Magenta coordinate:(214, 52, 1.50), Position: (2, 10)
Magenta coordinate:(230, 51, 0.00), Position: (2, 11)
Magenta coordinate:(50, 78, 8.10), Position: (3, 0)
Magenta coordinate:(68, 79, 9.02), Position: (3, 1)
Magenta coordinate:(86, 80, 9.57), Position: (3, 2)
Magenta coordinate:(103, 80, 9.95), Position: (3, 3)

Magenta coordinate:(120, 81, 10.49), Position: (3, 4)
Magenta coordinate:(138, 82, 11.41), Position: (3, 5)
Magenta coordinate:(184, 75, 6.12), Position: (3, 8)
Magenta coordinate:(200, 73, 5.33), Position: (3, 9)
Magenta coordinate:(214, 68, 1.03), Position: (3, 10)
Magenta coordinate:(230, 68, 0.00), Position: (3, 11)
Magenta coordinate:(248, 69, 0.00), Position: (3, 12)
Magenta coordinate:(50, 94, 8.65), Position: (4, 0)
Magenta coordinate:(68, 95, 9.02), Position: (4, 1)
Magenta coordinate:(86, 96, 9.02), Position: (4, 2)
Magenta coordinate:(102, 96, 9.02), Position: (4, 3)
Magenta coordinate:(120, 98, 10.30), Position: (4, 4)
Magenta coordinate:(137, 98, 10.49), Position: (4, 5)
Magenta coordinate:(154, 97, 10.41), Position: (4, 6)
Magenta coordinate:(184, 92, 6.63), Position: (4, 8)
Magenta coordinate:(200, 90, 3.75), Position: (4, 9)
Magenta coordinate:(230, 85, 0.00), Position: (4, 11)
Magenta coordinate:(68, 112, 7.73), Position: (5, 1)
Magenta coordinate:(85, 112, 9.03), Position: (5, 2)
Magenta coordinate:(102, 113, 9.35), Position: (5, 3)
Magenta coordinate:(120, 114, 9.41), Position: (5, 4)
Magenta coordinate:(136, 114, 9.38), Position: (5, 5)
Magenta coordinate:(184, 109, 6.12), Position: (5, 8)
Magenta coordinate:(200, 106, 3.75), Position: (5, 9)
Magenta coordinate:(213, 102, 0.66), Position: (5, 10)
Magenta coordinate:(230, 102, 0.00), Position: (5, 11)
Magenta coordinate:(247, 102, 0.00), Position: (5, 12)
Magenta coordinate:(49, 127, 6.63), Position: (6, 0)
Magenta coordinate:(68, 128, 8.37), Position: (6, 1)
Magenta coordinate:(84, 129, 8.82), Position: (6, 2)
Magenta coordinate:(102, 129, 8.98), Position: (6, 3)

Magenta coordinate:(118, 129, 10.12), Position: (6, 4)
Magenta coordinate:(136, 130, 9.02), Position: (6, 5)
Magenta coordinate:(168, 127, 6.63), Position: (6, 7)
Magenta coordinate:(198, 120, 1.48), Position: (6, 9)
Magenta coordinate:(212, 119, 0.00), Position: (6, 10)
Magenta coordinate:(230, 119, 0.66), Position: (6, 11)
Magenta coordinate:(247, 119, 0.94), Position: (6, 12)
Magenta coordinate:(49, 144, 7.56), Position: (7, 0)
Magenta coordinate:(67, 144, 7.05), Position: (7, 1)
Magenta coordinate:(84, 145, 8.04), Position: (7, 2)
Magenta coordinate:(102, 146, 8.49), Position: (7, 3)
Magenta coordinate:(118, 146, 8.49), Position: (7, 4)
Magenta coordinate:(213, 136, 0.94), Position: (7, 10)
Magenta coordinate:(246, 135, 0.94), Position: (7, 12)
Magenta coordinate:(48, 160, 6.63), Position: (8, 0)
Magenta coordinate:(69, 162, 7.19), Position: (8, 1)
Magenta coordinate:(84, 162, 7.38), Position: (8, 2)
Magenta coordinate:(101, 162, 8.06), Position: (8, 3)
Magenta coordinate:(118, 163, 8.92), Position: (8, 4)
Magenta coordinate:(136, 164, 9.02), Position: (8, 5)
Magenta coordinate:(246, 153, 0.94), Position: (8, 12)
Magenta coordinate:(84, 179, 8.49), Position: (9, 2)
Magenta coordinate:(101, 179, 7.56), Position: (9, 3)
Magenta coordinate:(118, 179, 8.49), Position: (9, 4)
Magenta coordinate:(152, 179, 7.99), Position: (9, 6)
Magenta coordinate:(184, 125, 5.11), Position: (9, 8)
Magenta coordinate:(212, 170, 0.63), Position: (9, 10)
Magenta coordinate:(230, 170, 0.66), Position: (9, 11)
Magenta coordinate:(247, 170, 0.42), Position: (9, 12)
Magenta coordinate:(196, 187, 0.66), Position: (10, 9)
Magenta coordinate:(212, 187, 0.00), Position: (10, 10)

Magenta coordinate:(230, 187, 0.66), Position: (10, 11)

Magenta coordinate:(247, 187, 0.94), Position: (10, 12)

Magenta coordinate:(42, 203, 0.00), Position: (11, 0)

Magenta coordinate:(195, 203, 1.48), Position: (11, 9)

Magenta coordinate:(216, 205, 0.73), Position: (11, 10)

Magenta coordinate:(230, 204, 0.00), Position: (11, 11)

Magenta coordinate:(247, 202, 1.10), Position: (11, 12)

where:

- "Blue coordinate" is a label of the UPPER_LEFT vertex of connected edges
- "Dark blue coordinate" is a label of the UPPER_RIGHT vertex of connected edges
- "Dark yellow coordinate" is a label of the LOWER_RIGHT vertex of connected edges
- "Magenta coordinate" is a label of the LOWER_LEFT vertex of connected edges
- "Position" is the position of the vertex on the PRMVS grid

Please refer to chapter 4, section 4 for more details on how to read and understand those extracted coordinates.

3D coordinates of extracted cylinder (chapter 5):

Blue coordinate:(196 21 0.00), Position: (1, 9)
Blue coordinate:(214 21 0.66), Position: (1, 10)
Blue coordinate:(231 21 0.00), Position: (1, 11)
Blue coordinate:(248 21 0.00), Position: (1, 12)
Blue coordinate:(87 52 11.41), Position: (2, 2)
Blue coordinate:(104 52 11.78), Position: (2, 3)
Blue coordinate:(122 51 12.30), Position: (2, 4)
Blue coordinate:(138 52 11.88), Position: (2, 5)
Blue coordinate:(154 51 11.02), Position: (2, 6)
Blue coordinate:(196 39 0.66), Position: (2, 9)
Blue coordinate:(214 39 0.48), Position: (2, 10)
Blue coordinate:(231 38 0.00), Position: (2, 11)
Blue coordinate:(249 38 0.38), Position: (2, 12)
Blue coordinate:(52 66 0.00), Position: (3, 0)
Blue coordinate:(68 71 8.86), Position: (3, 1)
Blue coordinate:(86 69 10.55), Position: (3, 2)
Blue coordinate:(104 69 11.41), Position: (3, 3)
Blue coordinate:(121 69 12.49), Position: (3, 4)
Blue coordinate:(137 70 11.60), Position: (3, 5)
Blue coordinate:(154 68 10.52), Position: (3, 6)
Blue coordinate:(170 68 9.57), Position: (3, 7)
Blue coordinate:(196 56 0.66), Position: (3, 9)
Blue coordinate:(214 55 0.00), Position: (3, 10)
Blue coordinate:(231 56 0.66), Position: (3, 11)
Blue coordinate:(248 55 0.00), Position: (3, 12)
Blue coordinate:(68 86 9.87), Position: (4, 1)
Blue coordinate:(86 86 10.78), Position: (4, 2)
Blue coordinate:(104 86 11.41), Position: (4, 3)

Blue coordinate:(120 87 11.95), Position: (4, 4)
Blue coordinate:(136 87 11.28), Position: (4, 5)
Blue coordinate:(154 85 10.49), Position: (4, 6)
Blue coordinate:(170 84 9.57), Position: (4, 7)
Blue coordinate:(196 74 1.33), Position: (4, 9)
Blue coordinate:(214 73 0.66), Position: (4, 10)
Blue coordinate:(231 73 0.66), Position: (4, 11)
Blue coordinate:(68 106 9.78), Position: (5, 1)
Blue coordinate:(86 103 10.64), Position: (5, 2)
Blue coordinate:(103 107 12.35), Position: (5, 3)
Blue coordinate:(120 103 11.44), Position: (5, 4)
Blue coordinate:(136 104 11.28), Position: (5, 5)
Blue coordinate:(154 102 10.49), Position: (5, 6)
Blue coordinate:(170 101 9.57), Position: (5, 7)
Blue coordinate:(196 91 0.66), Position: (5, 9)
Blue coordinate:(214 89 0.00), Position: (5, 10)
Blue coordinate:(231 90 0.66), Position: (5, 11)
Blue coordinate:(248 90 0.66), Position: (5, 12)
Blue coordinate:(69 119 9.57), Position: (6, 1)
Blue coordinate:(86 120 10.49), Position: (6, 2)
Blue coordinate:(103 122 12.18), Position: (6, 3)
Blue coordinate:(121 119 11.88), Position: (6, 4)
Blue coordinate:(137 119 10.49), Position: (6, 5)
Blue coordinate:(154 118 10.27), Position: (6, 6)
Blue coordinate:(170 118 9.43), Position: (6, 7)
Blue coordinate:(196 107 0.67), Position: (6, 9)
Blue coordinate:(214 107 0.66), Position: (6, 10)
Blue coordinate:(231 107 0.66), Position: (6, 11)
Blue coordinate:(248 107 0.00), Position: (6, 12)
Blue coordinate:(68 136 9.57), Position: (7, 1)
Blue coordinate:(88 136 10.88), Position: (7, 2)

Blue coordinate:(104 136 11.23), Position: (7, 3)
Blue coordinate:(121 136 10.88), Position: (7, 4)
Blue coordinate:(137 136 10.49), Position: (7, 5)
Blue coordinate:(154 135 9.87), Position: (7, 6)
Blue coordinate:(170 135 9.43), Position: (7, 7)
Blue coordinate:(230 126 0.66), Position: (7, 11)
Blue coordinate:(248 125 0.66), Position: (7, 12)
Blue coordinate:(68 155 10.28), Position: (8, 1)
Blue coordinate:(86 153 9.95), Position: (8, 2)
Blue coordinate:(104 153 10.88), Position: (8, 3)
Blue coordinate:(121 153 11.74), Position: (8, 4)
Blue coordinate:(137 153 10.95), Position: (8, 5)
Blue coordinate:(154 152 9.43), Position: (8, 6)
Blue coordinate:(170 152 9.57), Position: (8, 7)
Blue coordinate:(230 152 0.66), Position: (8, 11)
Blue coordinate:(68 170 9.57), Position: (9, 1)
Blue coordinate:(86 170 9.57), Position: (9, 2)
Blue coordinate:(104 170 10.88), Position: (9, 3)
Blue coordinate:(120 170 10.95), Position: (9, 4)
Blue coordinate:(137 170 10.95), Position: (9, 5)
Blue coordinate:(154 169 10.95), Position: (9, 6)
Blue coordinate:(170 169 9.57), Position: (9, 7)
Blue coordinate:(231 164 0.00), Position: (9, 11)
Blue coordinate:(249 160 0.48), Position: (9, 12)
Blue coordinate:(68 187 9.57), Position: (10, 1)
Blue coordinate:(87 187 10.36), Position: (10, 2)
Blue coordinate:(104 187 11.75), Position: (10, 3)
Blue coordinate:(120 187 11.79), Position: (10, 4)
Blue coordinate:(137 186 10.57), Position: (10, 5)
Blue coordinate:(154 186 8.92), Position: (10, 6)
Blue coordinate:(170 186 8.02), Position: (10, 7)

Blue coordinate:(214 176 0.66), Position: (10, 10)
Blue coordinate:(231 177 0.65), Position: (10, 11)
Blue coordinate:(68 205 9.04), Position: (11, 1)
Blue coordinate:(86 204 10.34), Position: (11, 2)
Blue coordinate:(102 207 11.43), Position: (11, 3)
Blue coordinate:(120 204 11.96), Position: (11, 4)
Blue coordinate:(137 204 10.34), Position: (11, 5)
Blue coordinate:(154 203 9.92), Position: (11, 6)
Blue coordinate:(170 203 8.49), Position: (11, 7)
Blue coordinate:(214 193 0.66), Position: (11, 10)

Dark blue coordinate:(205 21 0.66), Position: (1, 9)
Dark blue coordinate:(223 21 0.00), Position: (1, 10)
Dark blue coordinate:(240 21 0.73), Position: (1, 11)
Dark blue coordinate:(258 21 0.00), Position: (1, 12)
Dark blue coordinate:(95 52 11.36), Position: (2, 2)
Dark blue coordinate:(113 52 11.88), Position: (2, 3)
Dark blue coordinate:(128 51 12.26), Position: (2, 4)
Dark blue coordinate:(147 52 11.88), Position: (2, 5)
Dark blue coordinate:(163 51 10.95), Position: (2, 6)
Dark blue coordinate:(205 38 0.00), Position: (2, 9)
Dark blue coordinate:(223 39 0.00), Position: (2, 10)
Dark blue coordinate:(239 38 0.94), Position: (2, 11)
Dark blue coordinate:(256 38 0.94), Position: (2, 12)
Dark blue coordinate:(59 67 9.09), Position: (3, 0)
Dark blue coordinate:(79 69 9.85), Position: (3, 1)
Dark blue coordinate:(95 69 10.67), Position: (3, 2)
Dark blue coordinate:(113 69 12.36), Position: (3, 3)
Dark blue coordinate:(130 70 11.56), Position: (3, 4)
Dark blue coordinate:(147 69 11.41), Position: (3, 5)
Dark blue coordinate:(163 68 9.89), Position: (3, 6)

Dark blue coordinate:(178 66 8.65), Position: (3, 7)
Dark blue coordinate:(206 58 0.10), Position: (3, 9)
Dark blue coordinate:(223 55 0.00), Position: (3, 10)
Dark blue coordinate:(240 58 0.99), Position: (3, 11)
Dark blue coordinate:(257 55 0.66), Position: (3, 12)
Dark blue coordinate:(79 86 9.87), Position: (4, 1)
Dark blue coordinate:(96 86 10.78), Position: (4, 2)
Dark blue coordinate:(113 86 11.41), Position: (4, 3)
Dark blue coordinate:(130 87 11.43), Position: (4, 4)
Dark blue coordinate:(147 86 11.56), Position: (4, 5)
Dark blue coordinate:(163 85 7.99), Position: (4, 6)
Dark blue coordinate:(178 84 9.21), Position: (4, 7)
Dark blue coordinate:(206 77 3.38), Position: (4, 9)
Dark blue coordinate:(223 73 0.00), Position: (4, 10)
Dark blue coordinate:(240 73 0.66), Position: (4, 11)
Dark blue coordinate:(77 102 9.57), Position: (5, 1)
Dark blue coordinate:(96 103 10.43), Position: (5, 2)
Dark blue coordinate:(113 103 12.34), Position: (5, 3)
Dark blue coordinate:(130 103 11.95), Position: (5, 4)
Dark blue coordinate:(147 103 11.41), Position: (5, 5)
Dark blue coordinate:(163 102 9.95), Position: (5, 6)
Dark blue coordinate:(178 101 9.21), Position: (5, 7)
Dark blue coordinate:(206 92 2.10), Position: (5, 9)
Dark blue coordinate:(223 90 0.66), Position: (5, 10)
Dark blue coordinate:(239 90 0.94), Position: (5, 11)
Dark blue coordinate:(258 90 0.94), Position: (5, 12)
Dark blue coordinate:(79 119 10.36), Position: (6, 1)
Dark blue coordinate:(96 120 10.88), Position: (6, 2)
Dark blue coordinate:(113 120 12.16), Position: (6, 3)
Dark blue coordinate:(129 119 11.95), Position: (6, 4)
Dark blue coordinate:(146 119 10.49), Position: (6, 5)

Dark blue coordinate:(163 119 9.83), Position: (6, 6)
Dark blue coordinate:(178 119 9.21), Position: (6, 7)
Dark blue coordinate:(205 107 0.00), Position: (6, 9)
Dark blue coordinate:(223 106 0.00), Position: (6, 10)
Dark blue coordinate:(239 107 0.94), Position: (6, 11)
Dark blue coordinate:(257 107 0.00), Position: (6, 12)
Dark blue coordinate:(79 136 9.86), Position: (7, 1)
Dark blue coordinate:(95 136 10.49), Position: (7, 2)
Dark blue coordinate:(113 136 10.88), Position: (7, 3)
Dark blue coordinate:(130 137 10.95), Position: (7, 4)
Dark blue coordinate:(146 136 10.49), Position: (7, 5)
Dark blue coordinate:(163 136 9.47), Position: (7, 6)
Dark blue coordinate:(178 136 9.21), Position: (7, 7)
Dark blue coordinate:(240 128 3.32), Position: (7, 11)
Dark blue coordinate:(258 125 2.65), Position: (7, 12)
Dark blue coordinate:(79 153 10.80), Position: (8, 1)
Dark blue coordinate:(95 153 8.92), Position: (8, 2)
Dark blue coordinate:(113 153 10.95), Position: (8, 3)
Dark blue coordinate:(130 155 11.13), Position: (8, 4)
Dark blue coordinate:(146 153 10.95), Position: (8, 5)
Dark blue coordinate:(163 153 10.49), Position: (8, 6)
Dark blue coordinate:(178 152 8.18), Position: (8, 7)
Dark blue coordinate:(240 152 3.32), Position: (8, 11)
Dark blue coordinate:(78 171 9.95), Position: (9, 1)
Dark blue coordinate:(95 170 10.36), Position: (9, 2)
Dark blue coordinate:(113 170 10.80), Position: (9, 3)
Dark blue coordinate:(129 169 10.95), Position: (9, 4)
Dark blue coordinate:(146 170 9.95), Position: (9, 5)
Dark blue coordinate:(162 169 8.10), Position: (9, 6)
Dark blue coordinate:(178 169 6.83), Position: (9, 7)
Dark blue coordinate:(240 159 2.09), Position: (9, 11)

Dark blue coordinate:(259 159 0.94), Position: (9, 12)
Dark blue coordinate:(78 188 9.65), Position: (10, 1)
Dark blue coordinate:(95 186 10.56), Position: (10, 2)
Dark blue coordinate:(113 187 11. 32), Position: (10, 3)
Dark blue coordinate:(129 186 11.47), Position: (10, 4)
Dark blue coordinate:(146 187 10.49), Position: (10, 5)
Dark blue coordinate:(163 186 9.43), Position: (10, 6)
Dark blue coordinate:(179 186 6.57), Position: (10, 7)
Dark blue coordinate:(223 176 0.66), Position: (10, 10)
Dark blue coordinate:(241 177 0.00), Position: (10, 11)
Dark blue coordinate:(78 206 9.87), Position: (11, 1)
Dark blue coordinate:(95 204 9.96), Position: (11, 2)
Dark blue coordinate:(113 204 10.35), Position: (11, 3)
Dark blue coordinate:(129 203 10.98), Position: (11, 4)
Dark blue coordinate:(147 205 11.41), Position: (11, 5)
Dark blue coordinate:(163 204 9.95), Position: (11, 6)
Dark blue coordinate:(178 203 8.65), Position: (11, 7)
Dark blue coordinate:(221 193 1.48), Position: (11, 10)

Dark yellow coordinate:(206 27 1.10), Position: (1, 9)
Dark yellow coordinate:(223 29 0.66), Position: (1, 10)
Dark yellow coordinate:(241 26 1.24), Position: (1, 11)
Dark yellow coordinate:(258 29 0.42), Position: (1, 12)
Dark yellow coordinate:(95 60 10.49), Position: (2, 2)
Dark yellow coordinate:(113 60 10.88), Position: (2, 3)
Dark yellow coordinate:(129 60 10.96), Position: (2, 4)
Dark yellow coordinate:(145 60 10.12), Position: (2, 5)
Dark yellow coordinate:(161 59 9.57), Position: (2, 6)
Dark yellow coordinate:(205 46 0.66), Position: (2, 9)
Dark yellow coordinate:(223 46 0.94), Position: (2, 10)
Dark yellow coordinate:(240 43 0.99), Position: (2, 11)

Dark yellow coordinate:(258 45 0.94), Position: (2, 12)
Dark yellow coordinate:(60 71 9.46), Position: (3, 0)
Dark yellow coordinate:(79 76 10.88), Position: (3, 1)
Dark yellow coordinate:(96 76 10.97), Position: (3, 2)
Dark yellow coordinate:(113 77 11.43), Position: (3, 3)
Dark yellow coordinate:(129 77 11.49), Position: (3, 4)
Dark yellow coordinate:(147 76 10.36), Position: (3, 5)
Dark yellow coordinate:(161 76 9.45), Position: (3, 6)
Dark yellow coordinate:(178 74 8.10), Position: (3, 7)
Dark yellow coordinate:(206 64 0.66), Position: (3, 9)
Dark yellow coordinate:(223 64 0.00), Position: (3, 10)
Dark yellow coordinate:(241 61 1.48), Position: (3, 11)
Dark yellow coordinate:(258 63 0.66), Position: (3, 12)
Dark yellow coordinate:(96 93 10.36), Position: (4, 2)
Dark yellow coordinate:(130 93 10.36), Position: (4, 4)
Dark yellow coordinate:(146 93 9.95), Position: (4, 5)
Dark yellow coordinate:(163 92 9.86), Position: (4, 6)
Dark yellow coordinate:(178 91 8.10), Position: (4, 7)
Dark yellow coordinate:(206 78 2.10), Position: (4, 9)
Dark yellow coordinate:(223 80 0.94), Position: (4, 10)
Dark yellow coordinate:(240 79 1.48), Position: (4, 11)
Dark yellow coordinate:(79 109 9.86), Position: (5, 1)
Dark yellow coordinate:(96 109 9.86), Position: (5, 2)
Dark yellow coordinate:(113 110 10.36), Position: (5, 3)
Dark yellow coordinate:(130 110 10.48), Position: (5, 4)
Dark yellow coordinate:(146 110 9.43), Position: (5, 5)
Dark yellow coordinate:(163 109 9.21), Position: (5, 6)
Dark yellow coordinate:(178 108 8.10), Position: (5, 7)
Dark yellow coordinate:(205 98 0.00), Position: (5, 9)
Dark yellow coordinate:(223 98 0.66), Position: (5, 10)
Dark yellow coordinate:(239 98 0.00), Position: (5, 11)

Dark yellow coordinate:(258 92 0.03), Position: (5, 12)
Dark yellow coordinate:(79 126 10.32), Position: (6, 1)
Dark yellow coordinate:(96 126 9.86), Position: (6, 2)
Dark yellow coordinate:(113 127 10.36), Position: (6, 3)
Dark yellow coordinate:(130 126 9.43), Position: (6, 4)
Dark yellow coordinate:(146 127 9.95), Position: (6, 5)
Dark yellow coordinate:(163 126 9.20), Position: (6, 6)
Dark yellow coordinate:(206 115 0.00), Position: (6, 9)
Dark yellow coordinate:(223 115 0.00), Position: (6, 10)
Dark yellow coordinate:(240 115 0.00), Position: (6, 11)
Dark yellow coordinate:(258 115 0.00), Position: (6, 12)
Dark yellow coordinate:(79 143 9.86), Position: (7, 1)
Dark yellow coordinate:(95 144 9.95), Position: (7, 2)
Dark yellow coordinate:(113 144 10.36), Position: (7, 3)
Dark yellow coordinate:(129 144 9.95), Position: (7, 4)
Dark yellow coordinate:(145 144 9.57), Position: (7, 5)
Dark yellow coordinate:(163 142 9.10), Position: (7, 6)
Dark yellow coordinate:(240 133 0.66), Position: (7, 11)
Dark yellow coordinate:(258 132 0.66), Position: (7, 12)
Dark yellow coordinate:(78 160 9.43), Position: (8, 1)
Dark yellow coordinate:(95 161 9.95), Position: (8, 2)
Dark yellow coordinate:(113 161 10.36), Position: (8, 3)
Dark yellow coordinate:(129 161 11.86), Position: (8, 4)
Dark yellow coordinate:(146 160 9.43), Position: (8, 5)
Dark yellow coordinate:(163 159 8.92), Position: (8, 6)
Dark yellow coordinate:(177 159 7.73), Position: (8, 7)
Dark yellow coordinate:(78 177 9.43), Position: (9, 1)
Dark yellow coordinate:(95 178 10.36), Position: (9, 2)
Dark yellow coordinate:(112 178 9.95), Position: (9, 3)
Dark yellow coordinate:(129 178 9.95), Position: (9, 4)
Dark yellow coordinate:(146 177 9.43), Position: (9, 5)

Dark yellow coordinate:(161 177 8.65), Position: (9, 6)
Dark yellow coordinate:(177 176 8.62), Position: (9, 7)
Dark yellow coordinate:(241 167 0.00), Position: (9, 11)
Dark yellow coordinate:(259 166 0.48), Position: (9, 12)
Dark yellow coordinate:(78 192 8.44), Position: (10, 1)
Dark yellow coordinate:(95 195 10.36), Position: (10, 2)
Dark yellow coordinate:(112 195 10.49), Position: (10, 3)
Dark yellow coordinate:(129 195 9.67), Position: (10, 4)
Dark yellow coordinate:(146 194 9.43), Position: (10, 5)
Dark yellow coordinate:(163 193 8.92), Position: (10, 6)
Dark yellow coordinate:(178 193 8.10), Position: (10, 7)
Dark yellow coordinate:(223 184 0.00), Position: (10, 10)
Dark yellow coordinate:(241 183 0.66), Position: (10, 11)
Dark yellow coordinate:(78 207 9.24), Position: (11, 1)
Dark yellow coordinate:(95 212 9.42), Position: (11, 2)
Dark yellow coordinate:(112 212 9.36), Position: (11, 3)
Dark yellow coordinate:(129 212 9.21), Position: (11, 4)
Dark yellow coordinate:(146 211 8.92), Position: (11, 5)
Dark yellow coordinate:(163 211 8.92), Position: (11, 6)
Dark yellow coordinate:(177 210 7.17), Position: (11, 7)
Dark yellow coordinate:(223 201 0.66), Position: (11, 10)

Magenta coordinate:(196 29 0.94), Position: (1, 9)
Magenta coordinate:(214 29 0.00), Position: (1, 10)
Magenta coordinate:(230 28 0.66), Position: (1, 11)
Magenta coordinate:(248 29 0.00), Position: (1, 12)
Magenta coordinate:(87 60 10.88), Position: (2, 2)
Magenta coordinate:(104 61 10.88), Position: (2, 3)
Magenta coordinate:(120 60 10.49), Position: (2, 4)
Magenta coordinate:(138 60 10.36), Position: (2, 5)
Magenta coordinate:(154 60 10.49), Position: (2, 6)

Magenta coordinate:(196 46 0.66), Position: (2, 9)
Magenta coordinate:(214 47 0.00), Position: (2, 10)
Magenta coordinate:(230 46 0.66), Position: (2, 11)
Magenta coordinate:(248 46 0.00), Position: (2, 12)
Magenta coordinate:(53 74 10.23), Position: (3, 0)
Magenta coordinate:(70 76 10.36), Position: (3, 1)
Magenta coordinate:(87 77 10.40), Position: (3, 2)
Magenta coordinate:(104 77 10.49), Position: (3, 3)
Magenta coordinate:(120 77 10.57), Position: (3, 4)
Magenta coordinate:(136 76 9.57), Position: (3, 5)
Magenta coordinate:(154 76 9.22), Position: (3, 6)
Magenta coordinate:(170 75 9.02), Position: (3, 7)
Magenta coordinate:(196 64 0.62), Position: (3, 9)
Magenta coordinate:(214 64 0.00), Position: (3, 10)
Magenta coordinate:(230 63 0.94), Position: (3, 11)
Magenta coordinate:(248 63 0.00), Position: (3, 12)
Magenta coordinate:(86 93 9.02), Position: (4, 2)
Magenta coordinate:(120 94 8.90), Position: (4, 4)
Magenta coordinate:(136 93 9.57), Position: (4, 5)
Magenta coordinate:(154 93 9.95), Position: (4, 6)
Magenta coordinate:(170 92 9.02), Position: (4, 7)
Magenta coordinate:(196 81 0.00), Position: (4, 9)
Magenta coordinate:(214 81 0.00), Position: (4, 10)
Magenta coordinate:(230 81 0.48), Position: (4, 11)
Magenta coordinate:(69 109 8.92), Position: (5, 1)
Magenta coordinate:(86 110 9.95), Position: (5, 2)
Magenta coordinate:(103 110 10.54), Position: (5, 3)
Magenta coordinate:(120 110 10.36), Position: (5, 4)
Magenta coordinate:(137 110 9.43), Position: (5, 5)
Magenta coordinate:(154 110 9.49), Position: (5, 6)
Magenta coordinate:(170 109 8.02), Position: (5, 7)

Magenta coordinate:(196 98 0.00), Position: (5, 9)
Magenta coordinate:(214 98 0.00), Position: (5, 10)
Magenta coordinate:(230 98 0.35), Position: (5, 11)
Magenta coordinate:(248 98 0.00), Position: (5, 12)
Magenta coordinate:(68 124 7.99), Position: (6, 1)
Magenta coordinate:(86 127 9.57), Position: (6, 2)
Magenta coordinate:(103 124 9.82), Position: (6, 3)
Magenta coordinate:(120 127 9.75), Position: (6, 4)
Magenta coordinate:(138 127 9.66), Position: (6, 5)
Magenta coordinate:(154 127 8.59), Position: (6, 6)
Magenta coordinate:(196 115 0.00), Position: (6, 9)
Magenta coordinate:(214 115 0.00), Position: (6, 10)
Magenta coordinate:(231 115 0.00), Position: (6, 11)
Magenta coordinate:(248 115 0.00), Position: (6, 12)
Magenta coordinate:(69 143 9.02), Position: (7, 1)
Magenta coordinate:(86 144 9.02), Position: (7, 2)
Magenta coordinate:(103 143 9.92), Position: (7, 3)
Magenta coordinate:(120 144 10.49), Position: (7, 4)
Magenta coordinate:(138 144 9.86), Position: (7, 5)
Magenta coordinate:(154 144 9.32), Position: (7, 6)
Magenta coordinate:(230 132 0.16), Position: (7, 11)
Magenta coordinate:(248 132 0.00), Position: (7, 12)
Magenta coordinate:(69 160 9.43), Position: (8, 1)
Magenta coordinate:(86 161 9.95), Position: (8, 2)
Magenta coordinate:(104 161 9.86), Position: (8, 3)
Magenta coordinate:(120 161 9.95), Position: (8, 4)
Magenta coordinate:(137 160 9.92), Position: (8, 5)
Magenta coordinate:(154 161 9.02), Position: (8, 6)
Magenta coordinate:(170 161 8.57), Position: (8, 7)
Magenta coordinate:(69 177 8.92), Position: (9, 1)
Magenta coordinate:(86 178 9.95), Position: (9, 2)

Magenta coordinate:(104 178 10.36), Position: (9, 3)
Magenta coordinate:(120 178 9.95), Position: (9, 4)
Magenta coordinate:(136 177 9.02), Position: (9, 5)
Magenta coordinate:(154 177 8.49), Position: (9, 6)
Magenta coordinate:(170 176 8.49), Position: (9, 7)
Magenta coordinate:(232 167 0.00), Position: (9, 11)
Magenta coordinate:(248 165 0.48), Position: (9, 12)
Magenta coordinate:(68 194 8.49), Position: (10, 1)
Magenta coordinate:(86 195 9.02), Position: (10, 2)
Magenta coordinate:(104 195 9.42), Position: (10, 3)
Magenta coordinate:(120 195 10.02), Position: (10, 4)
Magenta coordinate:(137 194 8.10), Position: (10, 5)
Magenta coordinate:(154 194 7.17), Position: (10, 6)
Magenta coordinate:(170 193 7.10), Position: (10, 7)
Magenta coordinate:(214 184 0.06), Position: (10, 10)
Magenta coordinate:(231 183 0.00), Position: (10, 11)
Magenta coordinate:(68 211 8.32), Position: (11, 1)
Magenta coordinate:(86 212 8.47), Position: (11, 2)
Magenta coordinate:(102 209 9.08), Position: (11, 3)
Magenta coordinate:(120 212 9.47), Position: (11, 4)
Magenta coordinate:(137 211 8.49), Position: (11, 5)
Magenta coordinate:(154 211 8.92), Position: (11, 6)
Magenta coordinate:(170 210 7.56), Position: (11, 7)
Magenta coordinate:(214 201 0.17), Position: (11, 10)

where:

- "Blue coordinate" is a label of the UPPER_LEFT vertex of connected edges
- "Dark blue coordinate" is a label of the UPPER_RIGHT vertex of connected edges
- "Dark yellow coordinate" is a label of the LOWER_RIGHT vertex of connected edges

- "Magenta coordinate" is a label of the LOWER_LEFT vertex of connected edges
- "Position" is the position of the vertex on the PRMVS grid

Please refer to chapter 4, section 4 for more details on how to read and understand those extracted coordinates.